CHIP SPECIAL
Mensile di micro e personal computer
N° 11 Marzo 1986
L. 8000

Commodore 64 PROGRAMMI

Robot Computer

5

...nto postale-gruppo III/

Volare con il C. 64.
Le prese programmabili. Il calcolatore come robot da tavolo, giardiniere, portinaio, maggiordomo, pilota e light-jockey. Interfaccia per la computer-robotica.

tecniche nuove

Introduzione

Ci vuole proprio poco per portare l'home computer C64 alla realizzazione autosufficiente di un lavoro manuale. Con l'aiuto di uno della maggior parte dei brevi programmi presentati e di un semplice circuito aggiuntivo, il C64 può aiutarvi ad innaffiare i fiori, come antifurto, come animatore....
Ci siamo adoperati per limitare il dispendio di tempo necessario. Ad esempio nel simulatore di volo, che per il massimo realismo usa un telecomando originale da modellismo, si rinuncia alla grafica ad alta risoluzione.
Di questo volume fanno parte: per i fans del C64 interessati alle soluzioni pronte, una soluzione programmata da noi per la costruzione di un robot da tavolo con l'aiuto di una scatola di montaggio della Fisher-Technik. Per chi ha l'hobby dell'elettronica ci sono le "prese di corrente programmabili", un insieme di nuove possibilità per accendere, regolare e comandare.
Tutti i circuiti ed i listati proposti sono stati provati sperimentalmente. I principianti possono senza esitazioni riprodurre i circuiti, tenendo tuttavia le dita lontane dalle tensioni superiori ai 45 Volt. Ed ancora: non lavorate mai sotto tensione. Per la saldatura dei connettori, spegnete prima il calcolatore.
L'introduzione ai per nulla misteriosi "segreti" dell'autocostruzione dei circuiti facilita il nostro ingresso nel folto gruppo dei "costruttori dilettanti". Tutti i componenti utilizzati sono standard e possono perciò essere trovati facilmente ed a prezzi convenienti.
AugurandoVi buon divertimento nella trasformazione del vostro C64 in un home-robot.

La redazione di CHIP-SPECIAL

CHIP SPECIAL

mensile di programmi per micro e personal computer

Direttore responsabile: Giuseppe Nardella. Rivista mensile registrata il 16 giugno 1984 al numero 296 presso il Tribunale di Milano. Registro nazionale della stampa n. 630 v. 7 f. 233 - 16.12.82. **Distributore:** Messaggerie Periodici ME.PE., Via G. Carcano, 32 - 20141 Milano. **Stampa:** Arti Grafiche F.lli Jodice, Rosate (MI). **Fotocomposizione:** Grafica Quadrifoglio, Milano.
© per la Germania: Vogel Verlag, Würzburg; © per l'Italia: Tecniche Nuove, via Moscova 46/9A, 20121 Milano, tel. (02) 6590351, telex 334647 TECHS I. Spedizione in abbonamento postale gruppo III/70.

Desidero ricevere
un numero della rivista
(allego L. 1000 in
per contributo spese di

Nome
Cognome
Via
CAP
Città

CHIP

Mensile di micro e personal computer
20121 MILANO - VIA MOSCOVA 46/9A - TEL. 02/6590351

Indice

Utilità e cose importanti	4	Simboli grafici sullo schermo
Simulatore di volo	5	Il C64 impara a volare
Discoteca	11	Il C64 come animatore
Party	15	Il C64 come gioco di luce
Interruttori sonori	18	Il C64 come maggiordomo
Apriporta	22	Il C64 come portinaio
Pilotare	27	Il C64 come pilota
Sorveglianza delle stanze	29	Il C64 come angelo custode
Fiori	30	Il C64 come giardiniere
Amplificatore di segnali	32	Il C64 come controllore ferroviario
Per risparmiare corrente	34	Il C64 come sorvegliante
Collegamenti	39	Le prese programmabili
Laboratorio hobbistico	50	Il C64 come frequenzimetro
Attenuatori	51	Vi sia luce
Generatori di segnali	56	Vita da cani
Selezionare, ordinare	58	Il C64 come robot da tavolo
Alimentazione	73	Come aumentare la tensione del C64
Dilettanti	74	Come modificare il mio C64
Interfaccia	76	Introduzione alla computer-robotica

Simboli grafici sullo schermo

Nel vostro manuale non sono trattati i simboli grafici. Nei listati essi sono presentati come numeri o testi in parentesi graffe. Sullo schermo appaiono nel modo seguente:

Quando sul video appare il seguente simbolo		Significato
	SHIFT CLR/HOME	cancellare il video
	CLR/HOME	cursore all'inizio del video
	CTRL 9	reverse disattivato
	CTRL 0	reverse attivato
	SHIFT ↑ CRSR ↓	cursore in alto
	↑ CRSR ↓	cursore in basso
	SHIFT ← CRSR →	cursore a sinistra
	← CRSR →	cursore a destra
	CTRL 1	colore nero
	CTRL 2	colore bianco
	CTRL 3	colore rosso
	CTRL 4	colore turchese
	CTRL 5	colore violetto
	CTRL 6	colore verde
	CTRL 7	colore blu
	CTRL 8	colore giallo

Quando sul video appare il seguente simbolo		Significato
	C= 1	colore arancio
	C= 2	colore marrone
	C= 3	colore rosso chiaro
	C= 4	colore grigio 1
	C= 5	colore grigio 2
	C= 6	colore verde chiaro
	C= 7	colore blu chiaro
	C= 8	colore grigio 3
	CBM 1	liberamente definibile
	CBM 2	liberamente definibile
	CBM 3	liberamente definibile
	CBM 4	liberamente definibile
	CBM 5	liberamente definibile
	CBM 6	liberamente definibile
	CBM 7	liberamente definibile
	CBM 8	liberamente definibile

Il C64 impara a volare

Non soltanto gli appassionati di aereomodelli o i piloti possono imparare a volare, ma anche i possessori di un C64. Naturalmente l'aereoplano non è il computer, bensì un programma che rappresenta sullo schermo un apparecchio che può essere pilotato con un telecomando.
Consideriamo prima di tutto l'uscita del dispositivo per il telecomando. A questo scopo dobbiamo usare un oscilloscopio e possiamo verificare che vi è un segnale di una certa durata, seguito da sei impulsi più corti.
Utilizzeremo un usatissimo telecomando della Multiplex e precisamente il modello Europa. Questo dispositivo funziona secondo il cosiddetto principio PPM (Pulse-Position-Modulation) ovvero modulazione-impulso-posizione. Per il nostro aereoplano serviranno i canali due e tre. Il canale due serve per gli spostamenti verso il basso e verso l'alto, mentre il canale tre serve per dirigere l'aereoplano a destra ed a sinistra. Tutti gli altri segnali non verranno utilizzati. Riassumendo, impiegheremo un telecomando a due canali.
Per potere affrontare correttamente il problema, è necessario analizzare il segnale d'ingresso sulla User-port generato dal telecomando, questo segnale viene inviato all'ingresso PB0. Il range di uscita di un telecomando è superiore a 5 Volt.
Per questo motivo occorre regolare la tensione di uscita per un massimo di 5 Volt. Si usa a questo scopo un circuito integrato del tipo 74LS00 che ammette un ingresso massimo di 15 Volt. Come possiamo verificare su un oscilloscopio, gli impulsi modulati del trasmettitore non vengono alterati passando attraverso questo circuito e quindi la stessa frequenza degli impulsi di comando raggiunge l'ingresso della porta.

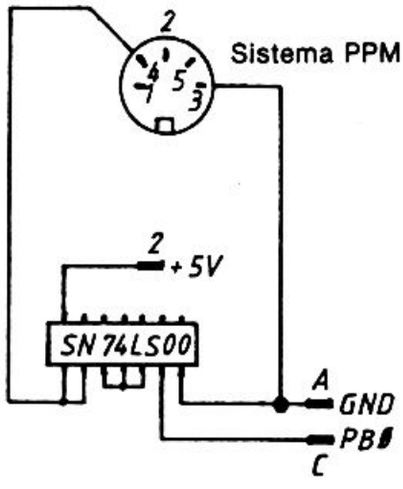


Fig. 1 - Interfaccia del C64 impara a volare.

Una corsa

Dato che sulla User-port del calcolatore giunge una serie di impulsi, si ricerca dapprima l'impulso più lungo. Questo impulso lungo serve realmente anche nei modelli come impulso di sincronizzazione. Gli impulsi individuali che seguono questo impulso di sincronizzazione, costituiscono i canali veri e propri. Attraverso la larghezza dell'impulso, si comunica al ricevitore di quanto occorre muovere il timone. Per noi che abbiamo un calcolatore, il primo compito è valutare la durata dell'impulso. Per prima cosa dobbiamo perciò identificare l'impulso di sincronizzazione. Una volta trovato, allora possiamo misurare il secondo impulso alto. Il valore così trovato ci fornisce un'informazione della larghezza di quest'impulso. Poiché tra il secondo ed il terzo canale deve esserci un segnale basso, noi possiamo, dopo l'arrivo di quest'impulso basso, misurare l'impulso alto seguente. Questo valore è correlato con lo spostamento a destra ed a sinistra dell'aereoplano. Gli impulsi seguenti non devono essere presi in considerazione dal nostro C64. Dopo di ciò, vi è a disposizione un breve tempo per spostare sullo schermo l'immagine dell'aereomodello. Si ricomincia quindi con la ricerca dell'impulso di sincronizzazione e la ripetizione dell'intero processo. Il programma descritto è rappresentato come diagramma di flusso nella figura 2 e verrà ora codificato passo per passo.

Un compromesso

Prima che vi accingiate ad affrontare piacevolmente questo compito, occorre dire che una soluzione a questo problema può essere trovata soltanto utilizzando il linguaggio di programmazione Assembler. Siccome questo linguaggio non è compreso da tutti, noi affronteremo accuratamente ogni passo del problema e descriveremo la sezione simbolica del programma. Con il listato seguente, potete inserire nel vostro C64 un programma BASIC che serve per caricare il programma in linguaggio macchina da richiamare in seguito con l'istruzione SYS.

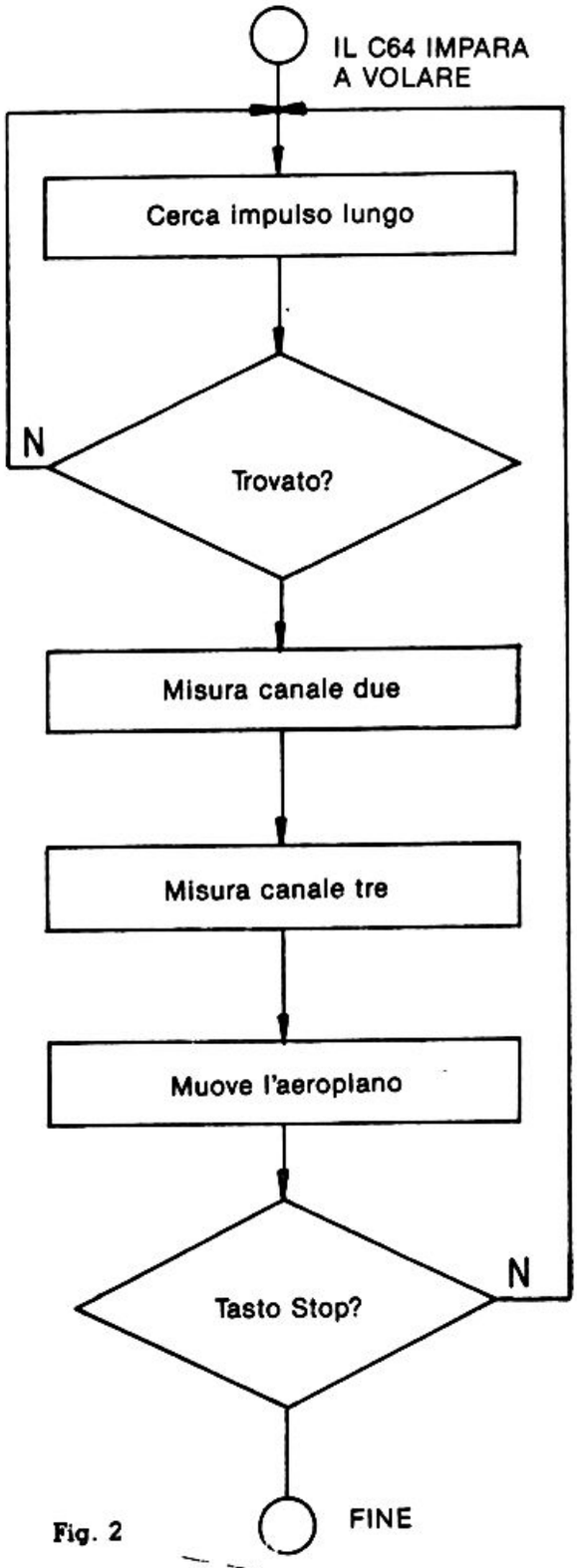


Fig. 2

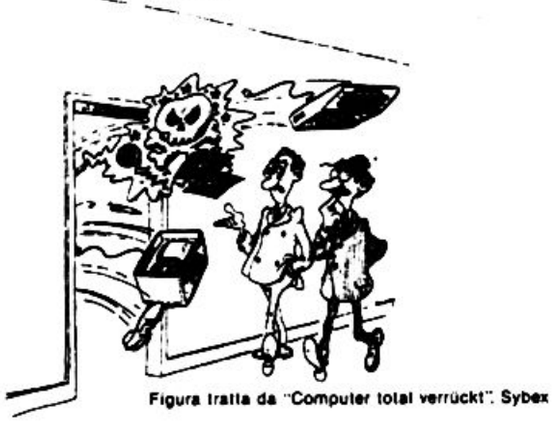


Figura tratta da "Computer total verrückt": Sybex

Il primo passo

Per analizzare il segnale di ingresso, occorre dapprima verificare se per il processo di misura della durata dell'impulso è sufficiente un contatore con valore massimo di 255. Un registro è costituito da 8 bit, perciò il numero massimo che può essere rappresentato è 255. Se la larghezza dell'impulso fosse maggiore di questo valore, dovremmo usare due registri per rappresentare questi numeri.
Il programma SIM1 presenta sullo schermo la corrispondente durata dell'impulso. Inserite quindi il listato che segue e lanciate il programma con il comando RUN. Sullo schermo comparirà una grande quantità di numeri che, in un primo momento, vi lascerà sicuramente delusi. Anche a noi è successa la stessa cosa, dato che ci aspettavamo una uscita numerica costante.

```
 5 REM SIM1
10 FOR I=49152TO49152+25
20 READ A:POKE I,A
30 NEXT
40 SYS 49152:PRINT PEEK(253):GOTO 40
50 DATA 120,169,0,133,253,133,254
60 DATA 173,1,221,201,255,208,249
70 DATA 230,253,173,1,221,201,255
80 DATA 240,247,88,96
```

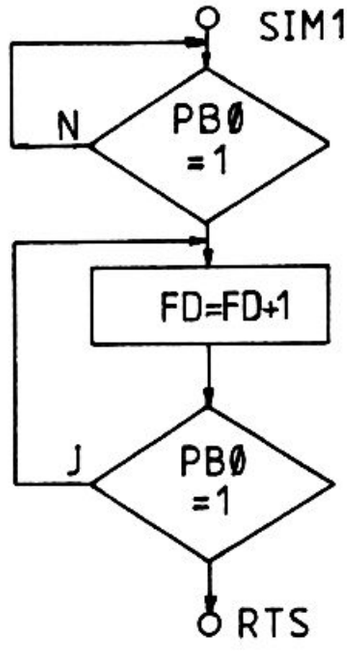


Fig. 3

Sicuramente possiamo riconoscere l'arrivo continuo di impulsi lunghi che vengono prodotti dagli impulsi di sincronizzazione. Siccome dopo ognu-

no di essi deve esserci un segnale basso, possiamo assegnarci come il compito di cercare dapprima questo impulso di sincronizzazione e quindi aspettare, in un altro ciclo di attesa, il secondo segnale d'ingresso alto. Quando questo segnale è arrivato, misuriamo la durata dell'impulso ed abbiamo perciò il valore corrispondente al secondo canale. Verifichiamo quindi il terzo passaggio per lo zero e misuriamo il canale numero tre. Ora possiamo tornare al programma relativo al movimento dell'aereoplano sullo schermo.
Inserite il listato seguente per la parte di programma SIM2. La parte di programma SIM1 viene perciò cancellata. SIM2 presenta sullo schermo soltanto gli impulsi lunghi. Date inizio al programma e considerate ora la serie di numeri. Tra un numero e l'altro si trovano i cosiddetti canali. Il prossimo compito è quindi trovare ed identificare questi canali per poterli poi misurare.

```
 5 REM SIM2
10 FOR I=49152 TO 49152+29
20 READ A:POKE I,A
30 NEXT
40 SYS 49152:PRINT PEEK(253):GOTO 40
50 DATA 120,169,0,133,253,173,1
60 DATA 221,201,255,208,249
70 DATA 230,253,173,1,221,201,255
80 DATA 240,247,165,253,201,130
90 DATA 144,230,88,96
```

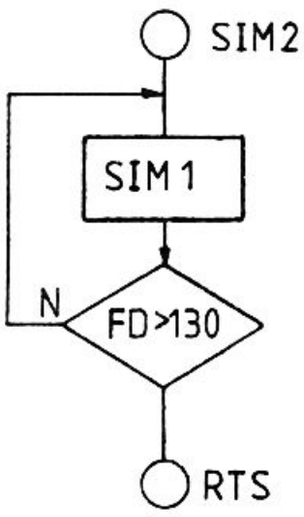


Fig. 4

Durante lo svolgimento del programma con la parte SIM2 potete già muovere la manopola del telecomando. Vedrete sullo schermo una sequenza di numeri che viene mostrata con velocità diversa. Possiamo quindi già osservare un piccolo risultato.

Chi cerca trova

La prossima parte del programma è illustrata nel diagramma a flusso SIM3 e viene ora analizzata. Qui dobbiamo determinare il movimento della leva di comando e cercare il corrispondente impulso. La leva di destra serve per il movimento verticale (su e giù) dell'aereo ed è quindi collegata con il canale due, mentre la leva di sinistra viene utilizzata per il movimento a destra e sinistra e corrisponde quindi al canale numero 3. Dopo che con il programma SIM2 è stato identificato l'impulso di sincronizzazione, occorre per prima cosa cercare il segnale basso successivo. Ciò avviene nel ciclo PB0=0. Il programma gira pertanto nel ciclo fintantoché il segnale 1 successivo, relativo al canale 2, non viene misurato sull'ingresso della porta PB0. Dopo che questo impulso è passato, si attende nuovamente, in un altro ciclo, il segnale 1 seguente. Dopo il segnale alto arriva il secondo segnale 0 che determiniamo con un altro ciclo. La richiesta successiva riguarda il segnale 1 seguente, relativo al canale 3, la cui durata viene ora determinata. Ciò avviene nel ciclo PB0=1 ed inoltre si incrementa di 1 la variabile FD. Il risultato di questa misura della durata dell'impulso viene poi conservata nella locazione FE.
In questo modo è stato misurato anche il canale 2 e possiamo dirigere il volo verso l'alto e verso il basso. Il canale seguente viene utilizzato per il movimento a destra ed a sinistra. Siccome tra i canali 2 e 3 si trova un segnale 0, occorre innanzitutto cercare questa condizione, utilizzando nuovamente il ciclo PB0=0. Non appena su PB0 giunge il segnale 1 del canale 3, allora dobbiamo nuovamente misurare questo canale. A questo scopo usiamo il ciclo PB0=1, FF=FF+1. In questo modo ha termine l'analisi dei canali, si ritorna dalla subroutine al programma principale ed il movimento dell'aereoplano può essere avviato. Per la rappresentazione delle nuvole è stato utilizzato un solo sprite. Inserite ora il listato definitivo ed avviate il programma con il comando RUN.
Nel diagramma a flusso relativo del programma SIM3 è nascosto un piccolo trucco. Come si è potuto verificare durante una prova, la singola determinazione della durata dell'impulso è relativamente imprecisa. Si possono trovare oscillazioni comprese tra 3 e 8 volte. L'aereoplano sullo schermo avrebbe perciò un aspetto tremolante; per ovviare a questo inconveniente abbiamo ripetuto il ciclo di misura di ogni canale per 4 volte utilizzando poi il valore medio. Nel diagramma a flusso questo accorgimento non è stato riportato, in quanto avrebbe soltanto creato confusione. Come potete osservare sullo schermo, il movimento dell'aereoplano è ancora molto lontano dal movimento reale di un modellino. Manca ad esempio l'effetto aereodinamico del modello; il prossimo passo con-

sisterà perciò nel migliorare la qualità dell'immagine volante.
A questo scopo abbiamo già iniziato una serie di prove e possiamo verificare che con l'aiuto di questo programma sono state poste le basi ed ora occorre soltanto generare una grafica per lo sfondo.
La rappresentazione delle nuvole, o di una catena di monti sullo sfondo, non consente di simulare il movimento dell'aereoplano in avanti quando questo sfondo si sposta sullo schermo.
L'illusione ottica del movimento dell'aereoplano può essere ottenuta solamente quando l'intera superficie dello schermo rappresenta uno sfondo che si muove continuamente in corrispondenza con il movimento dell'aereoplano. Solo in Assembler è possibile produrre una grafica di questo genere, infatti un programma in BASIC impiega circa 30 secondi per la formazione dell'immagine dello sfondo in movimento. Potete farvene un'idea considerando che la formazione di un panorama, con circa 64000 singoli punti per ogni immagine, non solo richiede una quantità enorme di dati, ma richiede anche un listato lunghissimo.
Il modellino volante deve inoltre modificare la sua immagine nei movimenti a destra e a sinistra. Ciò significa che occorre una rappresentazione tridimensionale che richiede un'altra subroutine in Assembler di notevole ampiezza. La soluzione di questo problema comporterebbe certamente un grande consumo di spazio nel programma e può essere superata, senza ulteriori difficolta, con l'aiuto del programma base. Detto semplicemente, se si rinuncia all'idea di un'immagine tridimensionale, si tratta di un semplice lavoro di diligenza.
Avete ora la possibilità di addentrarvi in questa problematica progettando il panorama corrispondente secondo i vostri desideri e le vostre necessità.

Fig. 5

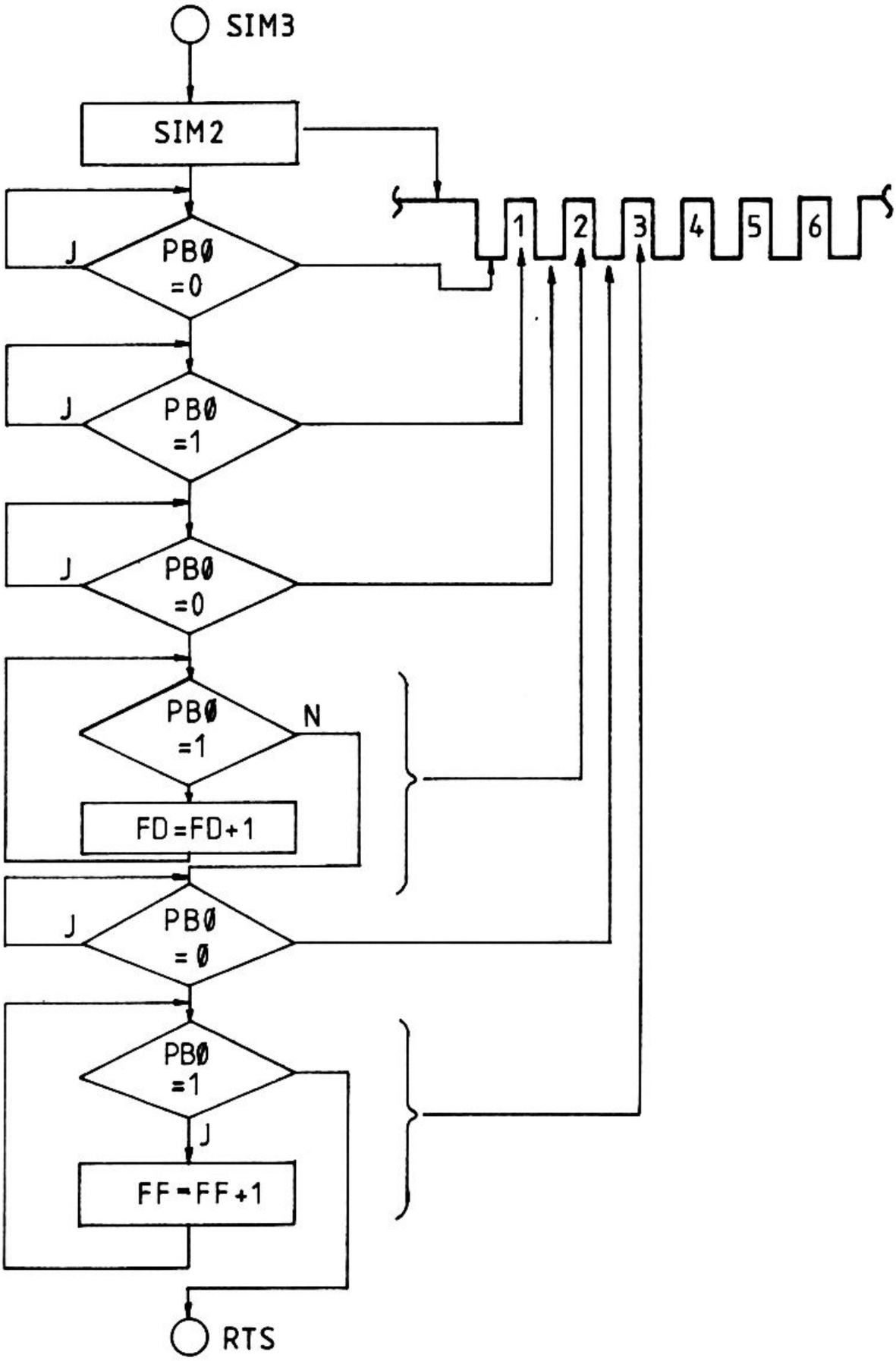

SIM3
SIM2
J
PBØ
=0
J
PBØ
=1
J
PBØ
=0
PBØ
=1
N
FD=FD+1
J
PBØ
=Ø
PBØ
=1
J
FF=FF+1
RTS
1
2
3
4
5
6

Fig. 6

```
10 REM C64 IMPARA A VOLARE
20 FORI=49152TO49349:READA:POKEI,A
30 Z=Z+A
40 NEXT
50 IFZ<>33506THEN PRINT"ERRORE NEI DATI":END
60 REM ATTIVAZIONE SPRITE
70 FORI=896TO1022:READA:POKEI,A
80 NEXT:V=53248
90 POKE V+32,9 :POKEV+33,6
100 PRINT CHR$(147)
110 PRINT CHR$(144)
120 POKEV+39,7:POKE V+40,7:POKEV+41,7
130 POKEV+21,15:POKE 2040,14
140 POKE2041,14:POKE2042,14
150 POKE2043,15
160 POKEV,70:POKEV+2,170:POKEV+16,4:POKEV+4,40
170 POKEV+6,150
180 POKEV+1,70:POKEV+3,100:POKEV+5,60
190 POKEV+7,170:POKEV+42,0
200 POKEV+23,8:POKEV+29,8
210 FORI=1TO25:PRINT"■":NEXT
220 X$="▬▬▬
```

```
1240 DATA 224,  4,208,237,165,252, 24
1250 DATA  74,133,252,165,251,106,133
1260 DATA 251,165,252, 24, 74,133,252
1270 DATA 165,251,106,133,251,165,251
1280 DATA 133,255, 88, 96
1295 REM DATI SPRITE DA 0 A 2
1300 DATA   0,  0,252,  0,  1,132
1310 DATA   0,  7,  3,  0, 12,  3
1320 DATA   0, 34,  2,  0, 32,  1
1330 DATA   0, 64,  1,  0, 32,  6
1340 DATA   0,192,  6,  0,128,  1
1350 DATA   3,  0,  2, 12,  0,  4
1360 DATA   8,  0,  2,112,  0,  1
1370 DATA 192,  0,  1,192,  0,  1
1380 DATA 128,  0,  1,128,  0,  3
1390 DATA 128,  0, 60,192,  0,  8
1400 DATA  63,255,224,  0
1500 REM DATI SPRITE3
1510 DATA   0,  0,  0,  0,  0,  0
1520 DATA   0,  0,  0,  0,  0,  0
1530 DATA   0,  0,  0,  0,126,  0
1540 DATA   0, 24,  0,  0, 24,  0
1550 DATA   0, 24,  0,255,255,255
1560 DATA  60, 60, 60,  1,231,128
1570 DATA   0,153,  0,  5,  0,160
1580 DATA   6,  0, 96,  4,  0, 32
1590 DATA   0,  0,  0,  0,  0,  0
1600 DATA   0,  0,  0,  0,  0,  0
1610 DATA   0,  0,  0
```

Il C64 come animatore

Gli appassionati delle discoteche conoscono sicuramente gli effetti luminosi che vengono generati sulle piste da ballo. La pista per discomusic è divisa in settori e sotto ogni settore si trovano delle lampade colorate che lampeggiano con un ordine determinato e rappresentano in tal modo una corsa di luci sulla pista. Abbiamo realizzato un circuito per effetti luminosi di questo tipo; mediante questo circuito potrete generare delle luci rotanti. Naturalmente non è necessario disporre di una discoteca ma potete ad esempio posizionare le lampade relative sul tavolo.

Il circuito

Per il collegamento delle lampade abbiamo progettato un cosiddetto interruttore di potenza che consiste di un relè elettromeccanico. Il relè utilizzato nel nostro circuito può sopportare, senza problemi, un carico di circa 6 Ampere. Poiché un relè di questo tipo con tensione di alimentazione di circa 5 Volt è difficile da trovare, abbiamo pensato di utilizzare un relè con tensione di alimentazione compresa tra gli 8 e i 12 Volt. Questa tensione di alimentazione deve essere ottenuta da un ge-

PB0	PB1	PB2	PB3	
1	2	3	4	PB4
5	6	7	8	PB5
9	10	11	12	PB6
13	14	15	16	PB7

Fig. 1

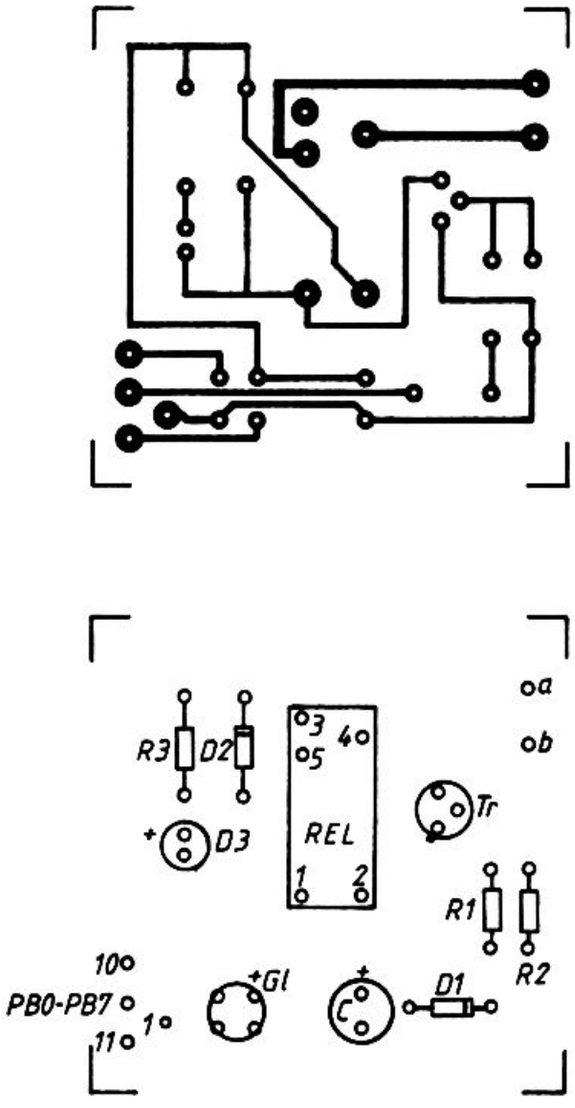
neratore esterno. Ovviamente, se avete a disposizione un generatore con queste caratteristiche, non dovrete preoccuparvi di costruirne uno.
Per il comando del relè è sufficiente un transistor del tipo BC140. Il diodo D2 funziona come diodo shunt di sicurezza per l'impulso induttivo prodotto alla chiusura ed all'apertura del relè. Sulla resistenza R1 può essere collegata qualunque uscita della User-port ed ogni segnale alto presente su questa uscita provocherà la chiusura del relè. In questo modo è possibile effettuare una commutazione di potenza con un relè senza che i due circuiti siano collegati elettricamente tra loro. L'interruttore di potenza descritto è quindi sconnesso dalla rete ed è perciò del tutto sicuro per il C64.
Per la suddivisione delle lampade, abbiamo ripartito la stanza in 16 quadrati: i 4 ingressi PB0-PB3 formano le linee verticali, mentre gli ingressi PB4-PB7 formano le righe orizzontali.
In questo modo è possibile, mediante la combinazione di due ingressi sulle righe orizzontali e verticali, individuare un punto ben determinato. In questo punto, o per meglio dire in questa area, si trova la lampada corrispondente che viene accesa per mezzo dell'interruttore di potenza. Con un relè da 6 Ampere è possibile collegare tranquillamente fino a 5 lampade da 60 Watt. Le uscite corrispondenti sono tutte collegate una all'altra attraverso porte AND e sono poi connesse con i singoli interruttori di potenza.
Per sicurezza, gli interruttori di potenza devono essere rinchiusi in una scatola in modo che non sia possibile toccare inavvertitamente la parte alimentata ad alta tensione.

Il programma

Dalla riga 10 alla riga 75 si provvede alla numerazione e suddivisione dell'area totale in 16 parti. Il comando delle lampade viene effettuato nelle subroutine 200 e 300 e viene simulato dall'accensione di un punto nell'area corrispondente. Per determinare i punti interessati, occorre determinare le uscite corrispondenti. Dato che la fantasia non ha limiti, il programma può essere realizzato anche con subroutine diverse. Nella subroutine 1000 viene determinato un ciclo di ritardo, mediante il contatore I, la cui durata può essere modificata a piacere.

Elenco dei componenti dell'interruttore di potenza

R1 = 1 KOhm
R2 = 10 KOhm
R3 = 330 Ohm
D1 = 1N4148
D2 = 1N4001
D3 = LED 5 mm
Tr = BC140
G1 = ponte raddrizzatore B40C800
C = 220 microF, 16V
Rel = Conrad 503363



Il C64 come animatore

Fig. 2

Interruttore di potenza

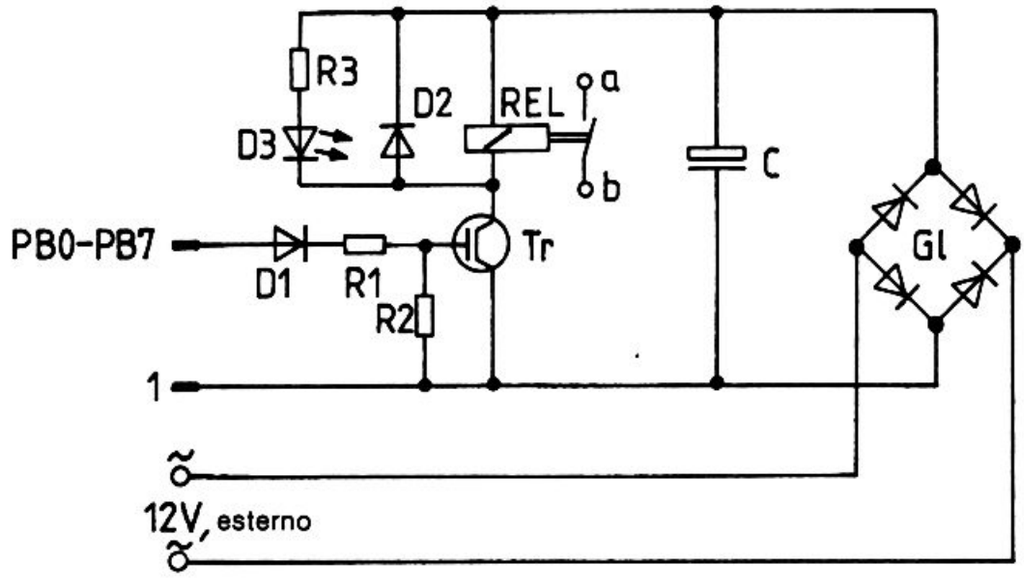


Fig. 3

```
5 REM C64 COME ANIMATORE
10 PO=56576:XN=1195:POKEPO+3,255
20 PRINT"■":PRINT:PRINT:PRINT"            ";
25 FORI=1TO9:READA:PRINTCHR$(A);:NEXT:PRINT
30 FORI=1TO9:READA:X$=X$+CHR$(A):NEXT
40 FORI=1TO9:READA:Y$=Y$+CHR$(A):NEXT
50 FORI=1TO3:PRINT"            ";X$:PRINT"            ";Y$:NEXT
60 PRINT"            ";X$:PRINT"            ";
70 FORI=1TO9:READA:PRINTCHR$(A);:NEXT
75 PRINT:PRINT"STOP = TASTO F1"
80 GOSUB 200:GOSUB 300
90 GETX$:IF X$=CHR$(133) THEN END
95 GOTO 80
100 DATA 176,99,178,99,178,99,178,99,174
110 DATA 98,32,98,32,98,32,98,32,98
120 DATA 171,99,123,99,123,99,123,99,179
130 DATA 173,99,177,99,177,99,177,99,189
200 REM PRIMO EFFETTO LUMINOSO
205 POKE PO+1,25:POKE XN,81 :POKE XN+54272,0:POKE XN+6,81:POKEXN+6+54272,0
210 GOSUB 1000
215 POKE XN,32:POKEXN+6,32
220 POKE XN+2+80,81:POKEXN+2+80+54272,0:POKEXN+4+80,81:POKEXN+4+80+54272,0
225 POKEPO+1,38:GOSUB 1000
230 POKEXN+2+80,32:POKEXN+4+80,32
235 POKEXN+2+160,81:POKEXN+2+160+54272,0:POKEXN+4+160,81:POKEXN+4+160+54272,0
240 POKEPO+1,70:GOSUB1000
245 POKEXN+2+160,32:POKEXN+4+160,32
250 POKEXN+240,81:POKEXN+6+240,81:POKEXN+240+54272,0:POKEXN+6+240+54272,0
255 POKEPO+1,137:GOSUB 1000
260 POKEXN+240,32:POKEXN+6+240,32
270 RETURN
300 REM SECONDO EFFETTO LUMINOSO
305 POKEXN+160,81:POKEXN+6+160,81:POKEPO+1,73:POKEXN+160+54272,0
310 POKEXN+6+160+54272,0:GOSUB 1000:POKEXN+160,32:POKEXN+6+160,32
315 POKEXN+80,81:POKEXN+80+54272,0:POKEXN+80+6,81:POKEXN+80+6+54272,0
320 POKEPO+1,41:GOSUB1000
325 POKEXN+80,32:POKEXN+80+6,32
340 RETURN
1000 FORI=1TO100:NEXT:RETURN
```

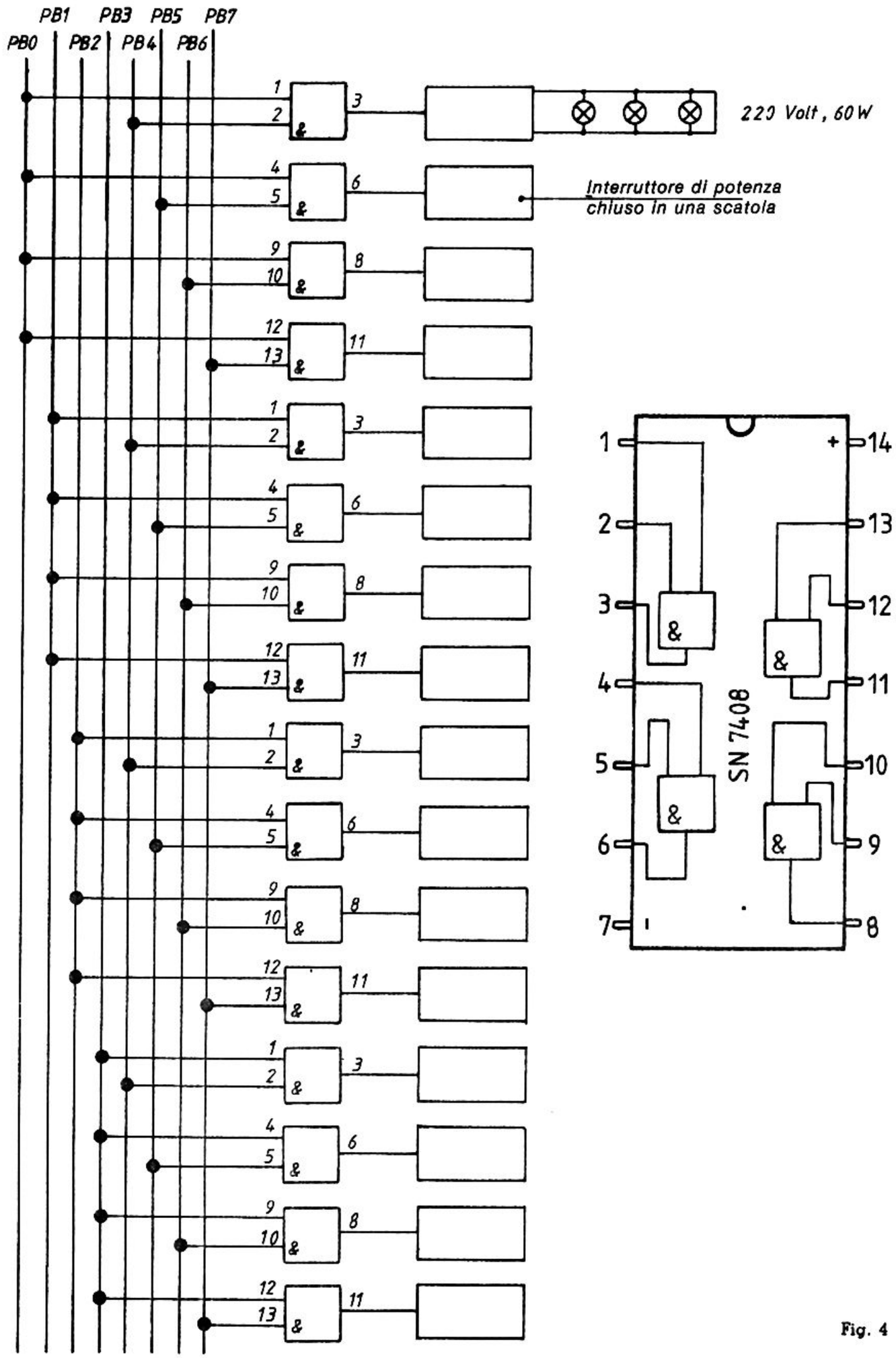

PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
220 Volt, 60 W
Interruttore di potenza chiuso in una scatola
SN 7408

Fig. 4

Il C64 come gioco di luce

L'interfaccia dai molteplici usi di ogni calcolatore è la User-port che è liberamente programmabile dall'utente. Come prima cosa, il programmatore determina la direzione dei dati che passano attraverso la porta. In tal modo ha la possibilità di far uscire dati attraverso la porta verso l'esterno oppure di fare entrare dati dall'esterno nella porta. Ciò costituisce il primo e più importante fondamento per il corretto utilizzo della User-port, dato che vi è il rischio che la porta sia programmata in uscita mentre è presente un segnale in ingresso. Questa situazione può provocare un cortocircuito ed eventualmente rovinare la User-port ed il relativo circuito integrato.
La seconda possibilità per il suo impiego consiste nel fatto che un programma, che è stato scritto per la User-port, può essere provato prima che il circuito elettrico di utilizzazione venga collegato con la User-port stessa. In talune circostanze, la programmazione della User-port può essere molto complessa, ad esempio quando i singoli ingressi devono essere commutati o come ingresso o come uscita. Per provare un programma così complicato, abbiamo progettato un programma di prova dedicato a questo compito.

Le caratteristiche richieste dalla User-port sono:

* Economicità e convenienza
* Semplicità della realizzazione elettronica
* Bidirezionalità del flusso di dati

L'interfaccia User-port che verrà descritta nel testo seguente, soddisfa queste condizioni. Essa verrà collegata alla User-port mediante un connettore, oppure il cavo di collegamento può anche essere saldato alla User-port stessa. Lo stato della porta viene costantemente mostrato sullo schermo dopo l'accensione del calcolatore. A questo scopo vi è la possibilità di condurre un segnale di ingresso dall'esterno verso il circuito di prova della User-port. I diodi Led vengono utilizzati per rappresentare lo stato degli ingressi.

L'interfaccia di prova

Poiché il circuito di uscita del calcolatore (CIA6526) è in grado di fornire in uscita soltanto una corrente limitata, è necessario, per pilotare i diodi Led, un amplificatore di potenza. Il circuito integrato qui impiegato, SN7416, contiene sei operazionali invertenti con uscita a collettore aperto che possono fornire una corrente massima di collettore di circa 40 mA. L'impedenza di ingresso dello stadio pilota è così alta che non è più possibile parlare di carico per la User-port. La tensione di alimentazione totale viene prelevata dal Pin 2 di uscita della User-port stessa e, mediante le resistenze da 330 Ohm, chiusa sui diodi Led.

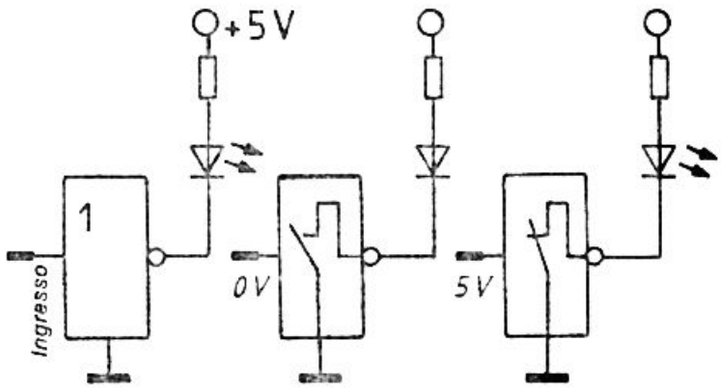


Fig. 1

Il circuito

Nella figura 1 viene schematizzato il funzionamento del circuito pilota che lavora sostanzialmente come un interruttore. Quando l'ingresso è collegato ad una tensione nulla, l'interruttore è aperto cosicché nessuna corrente può passare attraverso il diodo che rimane quindi spento. Quando invece l'ingresso è alimentato con una tensione positiva, allora, mediante una corrente di ingresso molto piccola, si chiude l'interruttore. Adesso può circolare una corrente attraverso il diodo Led che quindi si accende. Questo circuito funziona sia per i segnali in uscita dalla User-port che per i segnali che entrano nella User-port. Il circuito di prova della User-port rappresenta perciò un ampliamento della porta stessa, poiché le uscite della User-port sono a disposizione per altre funzioni logiche, senza che il collegamento con il circuito di prova della porta abbia un qualunque influsso sulla porta stessa o sul circuito esterno.

La costruzione

La realizzazione del circuito è relativamente semplice, trattandosi di pochi componenti. Ogni circuito integrato contiene sei stadi pilota, quattro dei quali vengono utilizzati. I numeri dei terminali forniscono le connessioni dei circuiti integrati. Nella realizzazione del circuito, occorre porre attenzione alla polarità dei diodi Led che vanno montati nel verso della conduzione, ovvero con l'anodo a +5 Volt. Ricordiamo che nei diodi Led, l'anodo corrisponde al terminale più lungo.

L'utilizzazione

Collegate ora il circuito di prova della User-port direttamente con un connettore oppure saldate direttamente il cavo sul circuito stampato del C64. Accendendo il calcolatore, potrete verificare che tutti i diodi Led sono accesi e la User-port è collegata in ingresso. Poiché nessun segnale negativo è collegato alla User-port, questa è automaticamente ad alta impedenza ed ogni ingresso assume valore logico 1. Per l'indirizzamento della User-port si usa un registro il cui indirizzo base è 56576 e che rappresenta lo stato della User-port. Per la direzione dei dati da leggere o da scrivere, è previsto l'uso del registro 56576+3. Questi registri contengono 8 bit ed ogni singola porta può essere perciò collegata come ingresso o come uscita. Se un bit è uguale a 0, allora la porta corrispondente è collegata in ingresso, viceversa se il bit è uguale a 1, la direzione dei dati è in uscita.

Corsa luminosa

Il registro dati per la porta B è collegato come uscita nel momento in cui introduciamo nel registro 56576 il valore 255. In questo modo infatti tutti i bit vengono messi a 1 ed il nostro C64 sa che tutte le porte della User-port sono programmate come uscite. Successivamente tutte le uscite della porta vengono messe a 0 con l'istruzione POKE 56576+1,0 cosicché il registro della porta B ha tutti i bit a 0 e tutti i nostri diodi Led si spengono. Per generare ora una sequenza di luci che si rincorrono, occorre mettere a 1 ogni singolo bit mettendo ogni volta a 0 il bit precedente. L'attribuzione dei valori ai singoli bit avviene mediante un'istruzione FOR che, usando il sistema binario, fornisce al registro il corrispondente numero binario.

Elenco dei componenti del circuito di prova della User-port:

IC1 = 2 x SN7416
LED = 8 x 5 mm
Resistenze = 8 x 330 Ohm
Connettore tipo MIL-C-21097 passo 3.96 mm

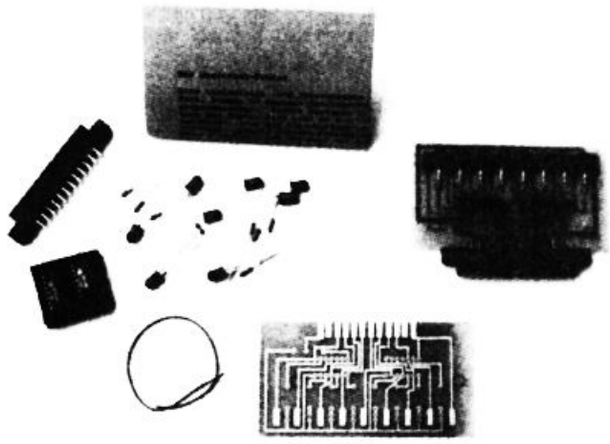


Fig. 2

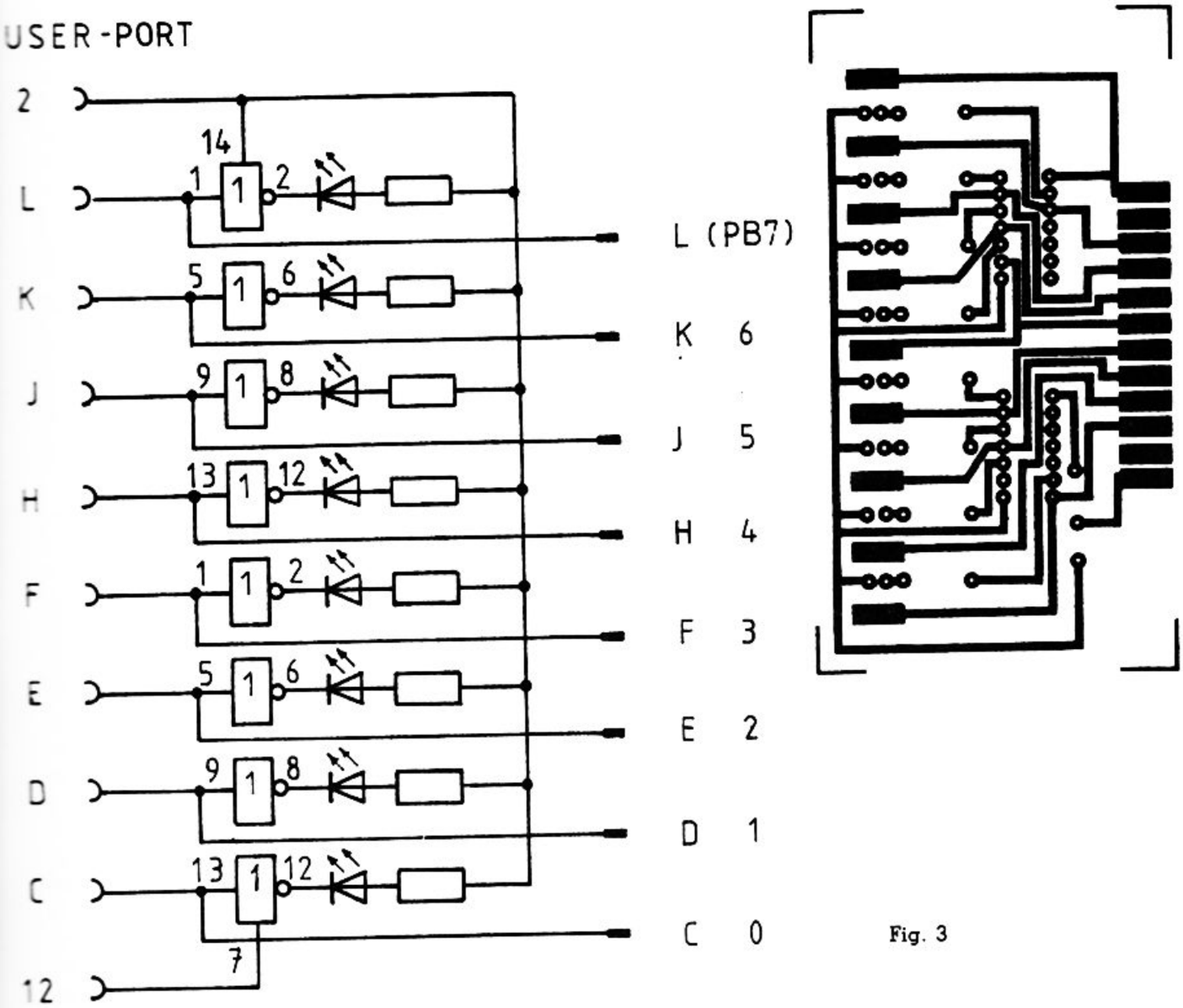


Fig. 3

```
10 REM GIOCO DI LUCE
20 PO=56576:REM PORTA B
30 POKEPO+3,255:REM PORTA B COME USCITA
40 PRINTCHR$(147):PRINT"F1 = STOP"
50 FORI=0TO7
60 GETX$:IFX$=CHR$(133)THENEND
70 POKE PO+1,2↑I:REM CODICE BINARIO
80 NEXT
90 FORI=7TO0STEP-1
100 POKE PO+1,2↑I:REM RITORNO
DEL CODICE BINARIO
110 NEXT
120 GOTO50
```

Figura tratta da "Computer total verrückt", Sybex

Il C64 come maggiordomo

Chi non conosce il celebre interruttore sonoro reperibile in tutti i cataloghi di prodotti per elettronici dilettanti? L'interruttore sonoro che adesso descriveremo è naturalmente adatto ad essere impiegato in connessione con un calcolatore. Il computer ha l'incarico di ricevere sulla User-port un segnale che proviene dall'interruttore sonoro. Vi è la possibilità di codificare questo segnale e l'interruttore sonoro diventa quindi un "Apriti, Sesamo".

Il circuito

Il primo ramo del ponte è costituito dalla resistenza R1 e dalla capsula microfonica a carbone che formano un partitore di tensione. In condizione di riposo, ovvero in assenza di rumori, il punto A si trova ad una tensione ben definita. Quando delle onde sonore arrivano al microfono, la tensione sale analogamente al livello sonoro. Un suono a volume alto provoca perciò un livello di tensione maggiore di un suono a volume basso. Il condensatore è caratterizzato da un'impedenza molto alta per la corrente continua e quindi impedisce che la tensione continua raggiunga l'ingresso 2. Esso permette perciò che soltanto le variazioni di tensione arrivino all'ingresso dell'amplificatore. Il secondo ramo del ponte è costituito dal partitore formato con le resistenze R3 e R2. Mediante questo partitore è possibile modificare la tensione sul punto B, variando quindi la sensibilità del circuito. La tensione di uscita dell'amplificatore operazionale è, in condizioni di riposo, di poco inferiore ai 5 Volt. Essa diventa subito negativa, non appena la tensione sull'ingresso 2 (partitore di tensione 1) diventa positiva rispetto all'ingresso 3 (partitore di tensione 2).
È facile comprendere che una tensione alta sul punto 3 fa sì che occorra una forte pressione sonora perché il potenziale sull'ingresso 2 salga a valori maggiori del potenziale fissato. In questo caso si forma allora, in uscita dall'amplificatore, un breve impulso negativo. Questo impulso viene utilizzato come trigger del successivo flip-flop monostabile, che ha il compito di mantenere per un tempo costante degli impulsi molto corti.
Descriviamo ora brevemente il funzionamento di un flip-flop monostabile. Questo flip-flop è caratterizzato da una sola (in greco mono vuol dire unico) condizione stabile. Quando sul suo ingresso giunge un impulso negativo, allora la sua uscita Q (pin 6) sale a 5 Volt e, dopo un tempo ben determinato, ritorna allo stato iniziale Q=0 Volt. Questo circuito è quindi in grado di generare impulsi di una determinata durata costante. La durata degli impulsi di ingresso non è correlata strettamente con la durata degli impulsi in uscita dal flip-flop che viene determinata dal condensatore C2.
Colleghiamo ora il circuito amplificatore con il flip-flop ed il nostro interruttore sonoro è pronto. Per provare e regolare la sensibilità si è anche pensato di usare un diodo Led. Fatto il collegamento con un cavo, si può poi procedere alla rotazione del trimmer potenziometrico R2 finché il diodo non si accende in seguito ad un rumore.

Il programma

Dato che la preparazione di un programma per l'interruttore sonoro è molto semplice, prendiamo in considerazione soltanto il conteggio dei rumori. Abbiamo utilizzato un contatore che, dopo la produzione del primo suono, conta i suoni per un tempo determinato. Se il numero corrisponde al numero previsto, allora l'uscita corrispondente viene attivata. Il programma di esempio che segue è stato scritto sicuramente per rumori complessi. Il codice rumore è costituito dal numero di impulsi sonori e dai loro intervalli temporali. Poiché è molto difficile per l'utente ripetere il codice sonoro già determinato con le stesse caratteristiche temporali, il codice sonoro da comparare con il codice dato deve essere considerato con una certa tolleranza. Questa è la parte più difficile di tutto il programma. Dalla linea 10 alla linea 95: questa parte del programma si occupa della costruzione dell'immagine sul monitor. Nella linea 20 si definisce la variabile PO=56576 che rappresenta l'indirizzo di ba-

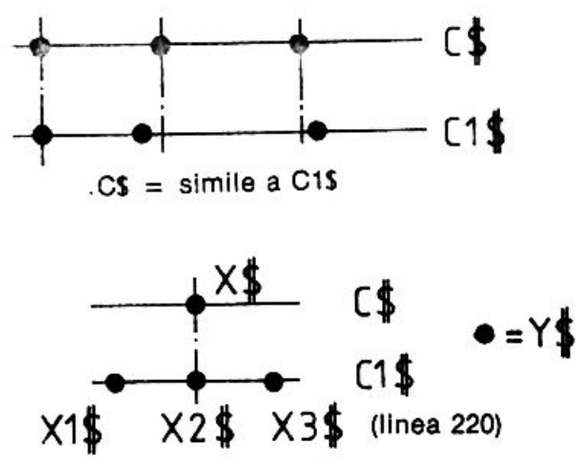


Fig. 1

se per la porta B. Nella linea 30, viene stabilita (con un POKE) la direzione dei dati in ingresso per la porta PB7 ed in uscita per tutte le altre porte.
Subroutine 300: questa subroutine serve per la realizzazione del codice sonoro che viene assegnato alla variabile C1$. Il codice sonoro viene, per così dire, introdotto sonoramente sull'interfaccia. Al primo segnale sonoro, si interrompe il ciclo senza fine alla linea 340 e si inizia un ciclo temporale con la variabile I=2 TO 41. L'utente può ora, all'interno di un determinato tempo, inserire una determinata serie di rumori. Sullo schermo, il suono viene rappresentato con un punto ed i tempi di pausa con un trattino. L'intero processo può quindi essere visualizzato sullo schermo. L'istruzione senza fine nella riga 390 blocca il programma fintantoché il segnale di ingresso sulla porta non ritorna a zero. In tal modo si può essere certi che anche un singolo rumore viene rappresentato con un solo punto sullo schermo. Nelle linee da 430 a 450 si può reintrodurre a piacere il codice rumore. C1$ rappresenta la regolazione della porta mentre C2$ permette che la porta venga azzerata nuovamente. L'inserimento di queste stringhe avviene nelle linee da 460 a 590. Questo processo corrisponde al processo già descritto per C1$.
Subroutine 100: se per C1$ o C2$ non deve essere determinato alcun codice, la linea 100 permette di tornare al menù principale. Nelle linee 140-190 viene introdotto il codice rumore. L'inizio dell'introduzione avviene con l'istruzione contenuta nella linea 140. Se non vi è alcun rumore, allora si torna alla linea 110. All'inizio di ogni codice rumore vi è sempre un punto dopo di che viene aperto il ciclo temporale già descritto e durante questo intervallo di tempo viene sorvegliato l'ingresso PB7 della porta. In corrispondenza al rumore introdotto, nelle linee 160-190, il valore viene assegnato a C$. Se C$ corrisponde a C1$ o a C2$, le linee 195 e 200 fanno saltare il programma direttamente all'azione.
In caso contrario, occorre controllare se esiste qualche somiglianza con C1$ o con C2$. Perciò, nella linea 210, la stringa C$ viene confrontata, mediante una variabile contatore, con la stringa C1$ presa come modello. Se in C$ vi è un punto uguale a Y$ e in C1$ si ha un punto Y$ immediatamente a sinistra, sotto o subito a destra, allora questo punto sonoro è simile al punto sonoro del codice. Sono anche accettabili due punti a sinistra o due

Fig. 2

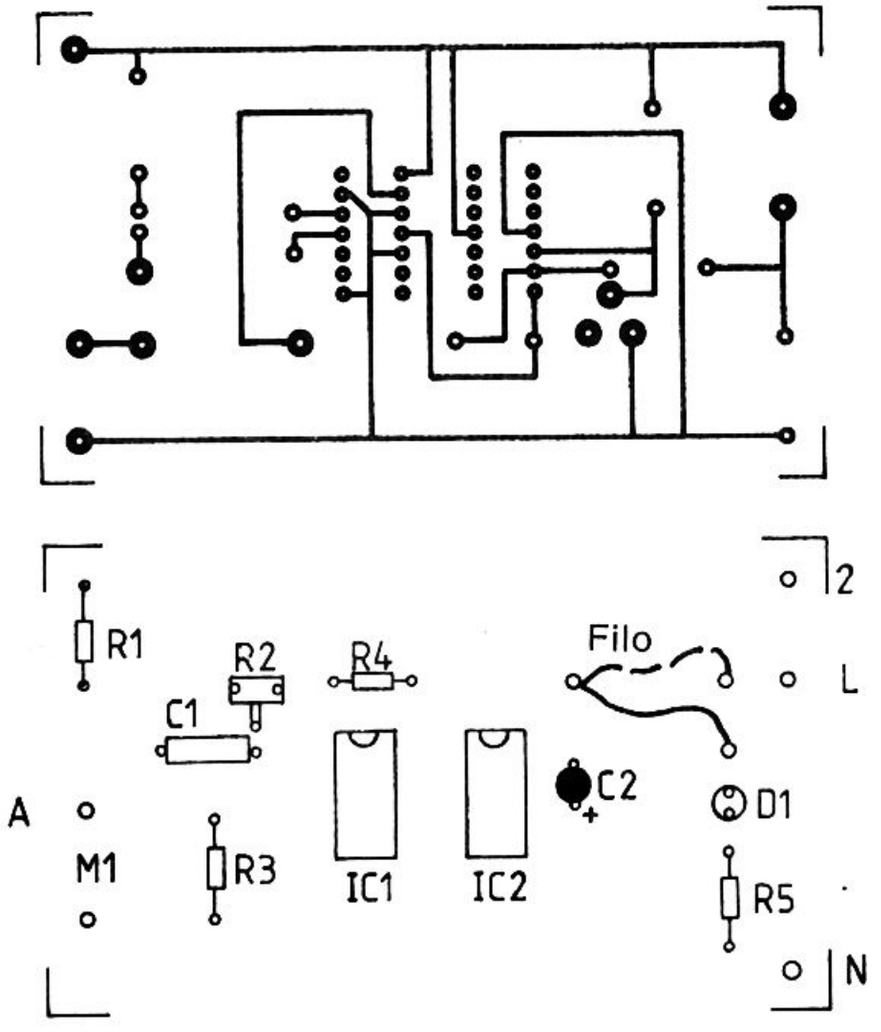


Fig. 3

punti a destra, ma in tal modo la sensibilità del comando è ridotta. Se C$ è in questo modo simile a C1$, allora alla variabile ausiliaria R2$ nella linea 240 non viene attribuito il valore "F". Nella linea 260 viene poi regolata la porta 1. Si ritorna quindi alla linea 110 e il prossimo ciclo di rumori può essere introdotto. Se C$ non è simile a C1$, allora nella linea 250 si salta alla 265 dove l'intero processo, già descritto per C1$, viene ripetuto nuovamente per quanto riguarda C2$. In tal modo termina il confronto di somiglianza di C$ con C1$ e C2$. Premendo il tasto F1, il programma termina la subroutine 100 alla riga 120 e ritorna al menù principale.

Elenco dei componenti

R1	=	1,5 KOhm
R2	=	10 KOhm
R3	=	3,9 KOhm
R4	=	1 MOhm
R5	=	330 Ohm
C1	=	100 nF
C2	=	100 microF
M1	=	Capsula microfonica a carbone
D1	=	Led verde, 5mm
IC1	=	LM324
IC2	=	SN74121

```
5 REM C64 COME MAGGIORDOMO
10 REM REGOLAZIONE DELLA USER PORT
20 PO=56576
30 POKEPO+3,255-128:POKEPO+1,0
40 PRINTCHR$(147)
50 PRINT"  *******>      MENU'      <*******
55 PRINT"  *                                *
60 PRINT"  * CODICE RUMORE ..... 1 *
65 PRINT"  *                                *
70 PRINT"  * INIZIO RUMORE ..... 2 *
75 PRINT"  *                                *
80 PRINT"  **********************************
85 GETX$:IFVAL(X$)<=0ORVAL(X$)>2THEN85
90 ONVAL(X$)GOSUB300,100
95 GOTO 40
100 IFC1$=""ORC2$=""THENPRINT"NESSUN CODICE DI PARAGONE":FORK=1TO1500:
NEXT:RETUR N
105 PRINTCHR$(147):PRINT"BATTETE PREGO!":PRINT:PRINT"F1 = MENU ":PRINT
110 GETX$:IFX$=""THEN130
120 IFX$=CHR$(133)THENRETURN
130 REM
140 IF(PEEK(PO+1)AND128)=0THEN110
150 C$="":Z=0:PRINTCHR$(113);:FORI=1TO40:IFZ=0THEN180
160 IF(PEEK(PO+1)AND128)>0THENC$=C$+CHR$(113):PRINTCHR$(113);:GOTO180
170 C$=C$+CHR$(99):PRINTCHR$(99);:GOTO190
180 IF(PEEK(PO+1)AND128)>0THEN180
190 Z=1:NEXT:C$=CHR$(113)+C$
195 IFC$=C1$THEN260
200 IFC$=C2$THEN295
210 Y$=CHR$(113):FORI=2TO41
220 X$=MID$(C$,I,1):X1$=MID$(C1$,I-1,1):X2$=MID$(C1$,I+1,1):X3$=MID$(C1$,I
230 IFX$<>Y$THEN250
235 IFX1$=Y$ORX2$=Y$ORX3$=Y$THEN250
240 I=43:R2$="F"
250 NEXT:IFR2$="F"THENR2$="":GOTO265
260 POKEPO+1,PEEK(PO+1)OR1:GOTO110
265 FORI=2TO41
270 X$=MID$(C$,I,1):X1$=MID$(C2$,I-1,1):X2$=MID$(C2$,I+1,1):X3$=MID$(C2$,I
275 IFX$<>Y$THEN290
280 IFX1$=Y$ORX2$=Y$ORX3$=Y$THEN290
285 I=43:R2$="F"
290 NEXT:IFR2$="F"THENR2$="":GOTO110
295 POKEPO+1,PEEK(PO+1)ANDNOT1:GOTO110
```

```
300 REM PREPARAZIONE CODICE RUMORE
310 PRINTCHR$(147):C1$=""
320 PRINT"CODICE RUMORE PER LA PORTA1=1 BATTITO!"
330 PRINT:FORK=1TO500:NEXT
340 IF(PEEK(PO+1)AND128)=0THEN340
350 Z=0:PRINTCHR$(113);:FORI=2TO41
360 IFZ=0THEN390
370 IF(PEEK(PO+1)AND128)>0THENC1$=C1$+CHR$(113):PRINTCHR$(113);:GOTO390
380 C1$=C1$+CHR$(99):PRINTCHR$(99);:GOTO400
390 IF(PEEK(PO+1)AND128)>0THEN390
400 Z=1:NEXT
410 C1$=CHR$(113)+C1$
420 PRINT
430 PRINT"CODICE OK?      S/N"
440 GETX$:IFX$=""THEN440
450 IFX$="N"THEN300
460 PRINT:C2$=""
465 PRINT"CODICE RUMORE PER LA PORTA1=0BATTITI!"
470 IF(PEEK(PO+1)AND128)=0THEN470
480 Z=0:PRINTCHR$(113);:FORI=2TO41
490 IFZ=0THEN520
500 IF(PEEK(PO+1)AND128)>0THENC2$=C2$+CHR$(113):PRINTCHR$(113);:GOTO520
510 C2$=C2$+CHR$(99):PRINTCHR$(99);:GOTO530
520 IF(PEEK(PO+1)AND128)>0THEN520
530 Z=1:NEXT
540 C2$=CHR$(113)+C2$
550 PRINT
560 PRINT"CODICE OK?      S/N"
570 GETX$:IFX$=""THEN570
580 IFX$="N"THEN460
590 RETURN
```

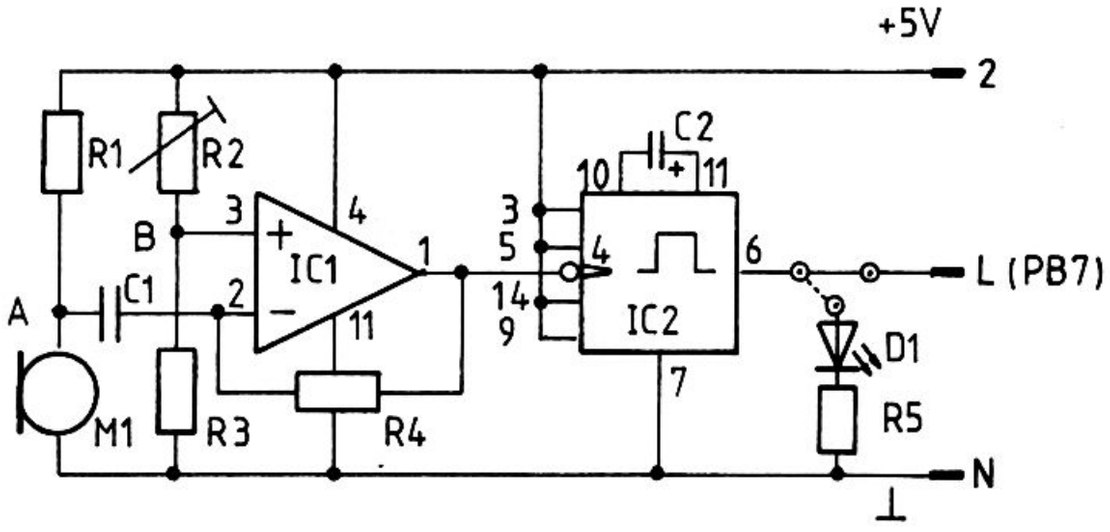


Fig. 4

Il C64 come portinaio

L'apertura di una porta sarà provocata dall'inserimento di un numero di codice segreto. Questo incarico, molto semplice e tuttavia efficace, può essere facilmente svolto dal C64 con un unico tasto. In seguito spiegheremo unicamente come introdurre dall'esterno un segnale nella User-port.
Il modo più facile e più conveniente per usare un segnale di ingresso su di un interruttore consiste nel collegamento con una resistenza.
L'alimentazione dell'interruttore a +5 Volt può essere ricavata dal pin 2 della User-port in quanto l'assorbimento totale del circuito non è eccessivo.
Se l'interruttore non è premuto, allora il segnale di ingresso nella User-port è nullo. Questo comportamento viene indicato anche con il nome di "logica positiva". Quando l'interruttore viene premuto, l'ingresso della porta deve diventare 1. La chiusura del contatto provoca un flusso di corrente da +5 Volt direttamente nell'ingresso mentre, se invece il contatto è aperto, l'ingresso della User-port è chiuso a massa dalla resistenza R.

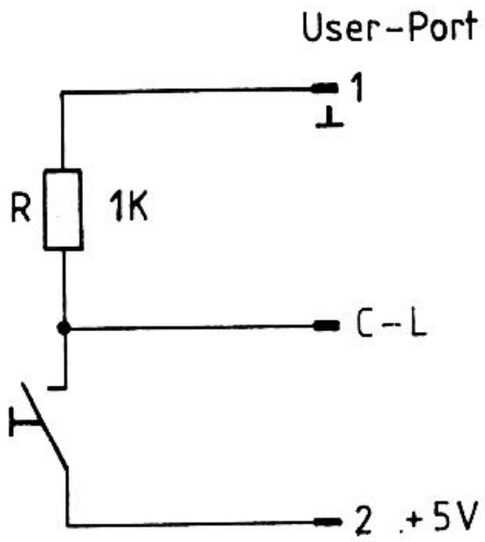


Fig. 1

La seconda possibilità che sull'ingresso si trovi un segnale 1 è data da un cosiddetto impulso di rimbalzo, infatti ogni interruttore meccanico provoca dei segnali di rimbalzo, sia in apertura che in chiusura dei contatti.
Mediante il collegamento di due porte Nand il segnale può assumere un solo stato. Il circuito presenta inoltre il vantaggio che i livelli di uscita della condizione logica (0 e 1) corrispondono agli stati della CIA6526. Con i circuiti integrati utilizzati, si realizza un flip-flop quadripolare al quale occorre aggiungere soltanto due resistenze. Il circuito utilizzato segue la cosiddetta logica 1-0.

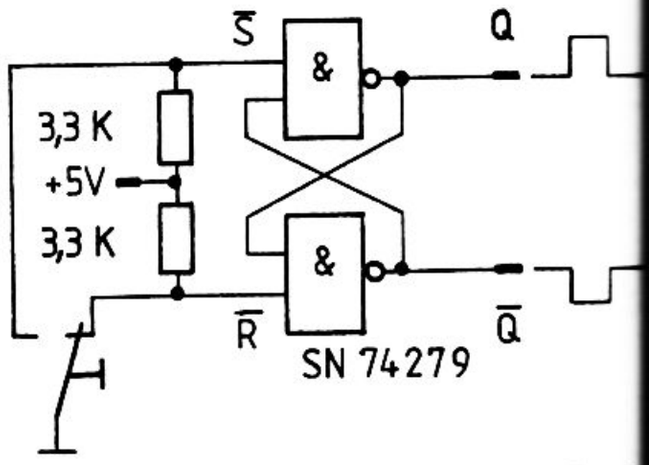


Fig. 2

Il numero di codice

Descriviamo ora brevemente l'impostazione del problema. La tastiera è composta in tutto da 10 interruttori ed è assicurata dalla parte esterna di una porta. Questi interruttori rappresentano le cifre da 0 a 9 e permettono in tal modo l'inserimento di una qualsiasi serie di numeri. Noi memorizziamo nel calcolatore un numero di codice, il quale verrà confrontato con il numero di codice inserito. Se il codice inserito corrisponde al codice depositato, allora viene inviato un segnale al dispositivo per l'apertura della porta.
La User-port viene quindi usata come circuito di ingresso, tranne una porta che è riservata come circuito di uscita per l'azionamento di un relè il quale serve, con il suo contatto, come interruttore di apertura della porta.

Il circuito

Consideriamo lo schema del circuito nel quale possiamo vedere i dieci tasti collegati ciascuno mediante una resistenza agli ingressi da PB0 a PB6. Se nessun tasto è premuto, allora su tutti gli ingressi vi è un segnale nullo. I tasti 7, 8 e 9 sono collegati, attraverso i diodi da D1 a D3, con il tasto 6. Premendo uno di questi tasti, il tasto 6 diventa attivo e sommandosi agli ingressi PB1-PB3, genera corrispondentemente i numeri 7, 8 e 9.
I diodi servono per l'accoppiamento con i tasti 1, 2 e 3. Il segnale di uscita PB7 serve per pilotare il relè.

Il programma

Proseguiamo con la descrizione del modo di lavo-

ro del programma, con l'aiuto del diagramma di flusso. In uscita dal menu, vi è la possibiltà di inserimento del codice numerico ed inizio del programma. La pressione del tasto 1 provoca il salto del programma alla subroutine 200 nella quale viene introdotto e controllato il numero di codice. Possono essere introdotti in tutto cinque caratteri a piacere, purché numerici. Dopo l'introduzione del codice numerico ed il controllo della sua correttezza, si ritorna al menù.
Premendo il tasto 2 inizia il programma. Vi è la possibilità di tornare al menù mediante la pressione del tasto funzione F1; per interrompere invece il programma occorre premere il tasto RUN/STOP. Proseguendo, il calcolatore controlla continuamente se è stato premuto qualche tasto, nel qual caso controlla se è stato oltrepassato un tempo massimo. Questo tempo serve pr il controllo del tempo d'inserimento totale. Se uno dei tasti è stato premuto, allora è previsto un tempo massimo di 5 secondi per l'introduzione del numero di codice. Nel caso che questo tempo totale venga superato, si ha un'interruzione di 1 minuto durante la quale non è più possibile introdurre alcun numero di codice e quindi aprire la porta.
È inoltre necessaria un'altra interruzione supplementare. Se viene premuto un tasto, occorre che venga rilasciato prima che sia trascorso un secondo. In questo modo ci si assicura che l'inserimento del numero di codice esatto avvenga in maniera ordinata. Dopo la pressione di un tasto, il numero così introdotto viene ordinato in una parola di confronto. Se questa parola di confronto è composta di 5 o più cifre e non corrisponde al codice inserito, allora il programma si interrompe e inizia il tempo di attesa di 1 minuto.
Dopo l'inserimento del numero di codice esatto e l'arresto del tempo relativo, l'uscita PB7 diventa attiva e la porta si apre. Si ripete quindi nuovamente il ciclo già descritto.

Il listato

Nelle linee dalla 5 alla 95 viene costruito il menù. Inoltre nella linea 10 viene assegnata la variabile PO che rappresenta l'indirizzo di base della porta. In seguito alla pressione dei tasti 1 o 2, si salta alle subroutine 100 o 300 rispettivamente. Cominciamo con la subroutine 100.
Nella linea 130 si richiede l'introduzione di C$ per mezzo dell'istruzione INPUT. Segue quindi un controllo sulla lunghezza di C$ e se questa lunghezza non oltrepassa i cinque caratteri, allora nelle linee da 140 a 180 vengono controllati i caratteri di

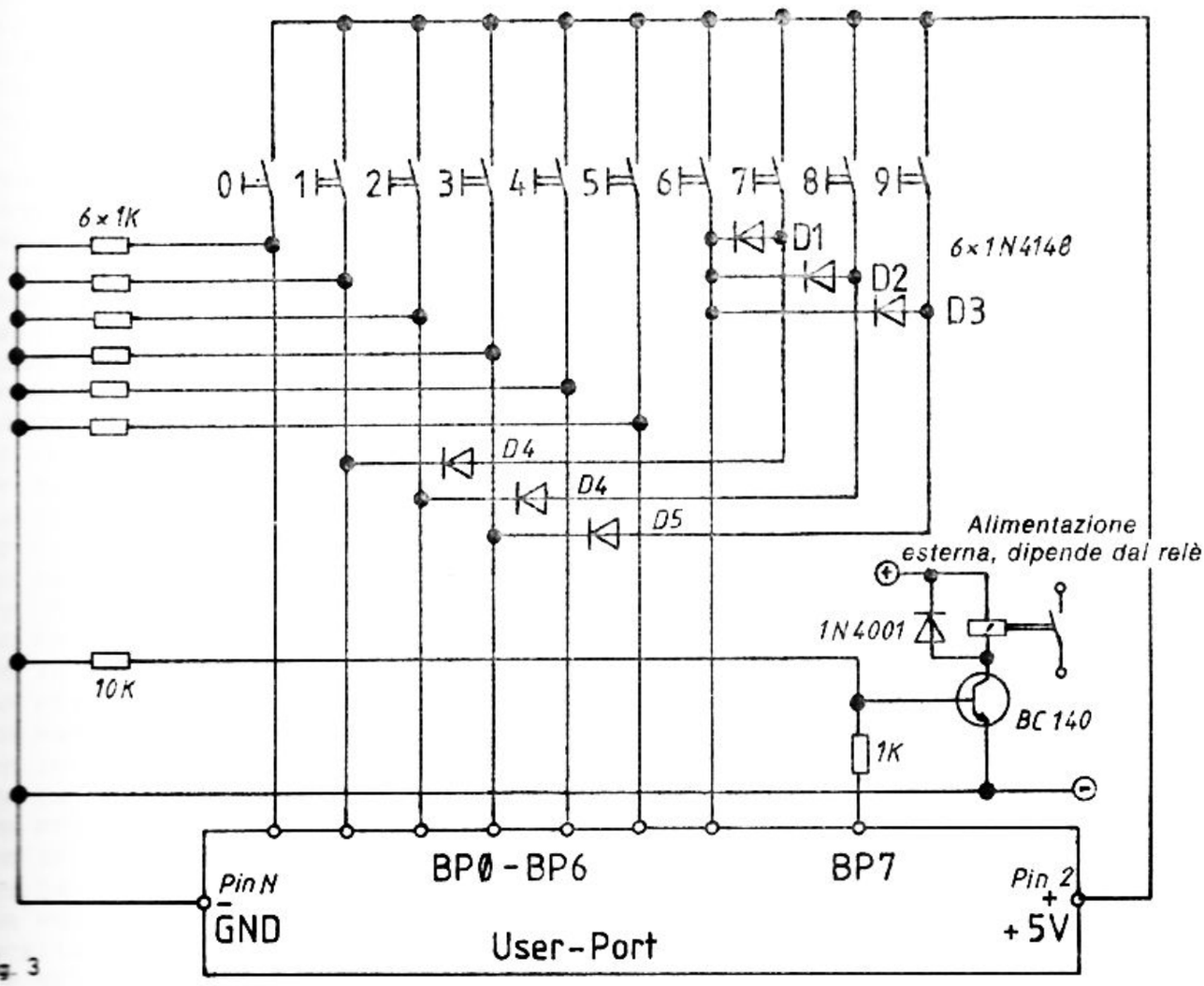


Fig. 3

C$. Se C$ è composta soltanto di numeri, allora non viene assegnata la variabile R2$ nella linea 170. In tal caso si salta la linea 190 e nella linea 200 si procede alla visualizzazione sullo schermo del codice inserito. La subroutine 100 ha così termine e si ritorna al menù.
La subroutine 300 è incaricata del comando della User-port. Si inizia con la linea 310 nella quale viene memorizzato lo stato della porta e viene assegnata la variabile Y. L'istruzione AND è neccessaria poiché la porta PB7 non deve essere utilizzata come ingresso, bensì come uscita. La subroutine 700 è usata per il protocollo di visualizzazione. Dopo che lo stato della porta è stato assegnato alla variabile Y, la linea 340 controlla se Y è nullo. In tal caso, possiamo tornare alla linea 300 e si ripete il controllo dello stato della porta. In caso contrario, il timer TI$ viene azzerato. La variabile MA viene posta uguale a 1, indicando con ciò che il timer nel prossimo ciclo non deve essere azzerato. Nelle linee 370-390 si controlla se X supera il valore 5, la variabile X viene generata nella linea 360 usando le ultime due cifre del timer TI$. Se quindi sono passati più di cinque secondi, nella linea 370 si azzerano sia MA sia i caratteri già introdotti. Anche il timer viene azzerato e, nella linea 380, si ha il tempo d'interruzione di un minuto. All'interno di questo tempo, il programma salta continuamente alle subroutine 600 e 700. Questi salti sono importanti per poter osservare sullo schermo in quale stato si trova con precisione il contatore. Trascorsi 59 secondi, il ciclo viene interrotto e si inizia nuovamente con la linea 300.
La subroutine 600 controlla la pressione del tasto funzione F1. Se questo tasto viene premuto, la variabile R2$ viene posta uguale a "R". In tal modo si può sapere che il tasto è già stato premuto e, nella linea 385, si salta alla linea 330.
Da ciò deduciamo che il tempo totale non ha ancora oltrepassato i cinque secondi. Nelle linee da 395 a 475 viene controllato l'ingresso della porta e lo stato esistente viene assegnato ad una determinata cifra. Quest'assegnazione avviene per mezzo della variabile ZI$. Nella linea 480, tutti i singoli caratteri vengono riuniti nella variabile EI$. Dopo la pressione di un tasto, esso deve essere rilasciato, prima che venga inserito il prossimo numero. Ciò è importante poiché altrimenti alla pressione di un tasto verrebbe conservato continuamente lo stesso numero. Questo controllo avviene nella linea 490. Se in ogni secondo non viene premuto nessun altro tasto, allora la linea 510 fa saltare alla linea 370. Alla variabile X viene assegnato il valore 6, simulando in tal modo l'oltrepassamento del tempo massimo. L'introduzione è corretta, come viene controllato nella linea 540, se la stringa EI$ appena inserita corrisponde a C$. In questo caso l'uscita PB7 viene posta uguale a 1 e con il ciclo della variabile K, compresa tra 1 e 50, viene assegnato un tempo per l'apertura della porta. Nel frattempo si viene rinviati alla subroutine 700 dove si provvede ad indicare sullo schermo lo stato corrispondente. Se il tempo di apertura è terminato, allora nella linea 550 si azzera l'uscita PB7 e, ponendo X=6, si salta alla linea 370 dove inizia il tempo di attesa di un minuto.
La subroutine alla linea 700 è relativa al protocollo di visualizzazione sullo schermo dove viene indicato se il portinaio è presente o no. Durante il tempo di attesa di un minuto, il portiere è assente e l'apertura della porta non è perciò possibile. Dopo il tempo di interruzione, viene indicato sullo schermo il tempo trascorso quando un tasto è premuto ed anche l'apertura della porta. In questo modo è possibile controllare sullo schermo l'intero procedimento.
Questo circuito, semplice da comprendere, insieme al programma relativo, assolve ad un importante incarico di sorveglianza che resta in funzione con sicurezza fintantoché non viene a mancare la corrente. Se la tensione manca, ovviamente il calcolatore si spegne ed il nostro portinaio può rientrare in servizio soltanto caricando nuovamente il relativo programma e reinserendo il numero di codice. Ma chi vuole vivere in mezzo ai robot deve anche essere pronto agli incidenti.

```
5 REM ****** C64 COME PORTINAIO ********
10 PO=56576
20 POKE PO+3,128:POKE 53248+32,6:POKE53248+33,6:PRINT CHR$(144)
30 PRINTCHR$(147)
40 PRINT"MENU":PRINT
50 PRINT"DETERMINAZIONE COD.";CHR$(158);"1";CHR$(144)
60 PRINT:PRINT"INIZIO PROGRAMMA ..";CHR$(158);"2";CHR$(144)
70 GET X$:IF X$="" THEN 70
80 IF X$="1" THEN GOSUB 100:GOTO30
90 IF X$="2" THEN PRINT CHR$(147):GOSUB 300:GOTO30
95 GOTO 70
100 REM DETERMINAZIONE DEL CODICE ******
110 PRINTCHR$(147):PRINT"INSERITE IL CODICE!"
120 PRINT:PRINT"MASSIMO 5 CARATTERI ";
130 INPUT C$:IF LEN(C$)>5 THEN 100
140 FOR I=1TO LEN(C$)
```

```
150 X=ASC(MID$(C$,I,1))
160 IF X>57 OR X<48 THEN PRINT:PRINT"INSERITE SOLO NUMERI!"
170 IF X>57 OR X<48 THEN FORK=1TO2000:NEXT:R2$="NUOVO":I=LEN(C$)+1
180 NEXT
190 IF R2$="NUOVO"THEN R2$="":GOTO 100
200 PRINT:PRINT"CODICE SALVATO!"
210 FORI=1TO1000:NEXT
220 RETURN
230 :
240 :
250 :
300 REM INIZIO PROGRAMMA ***************
310 Y=PEEK(PO+1)AND127 :I=0:GOSUB700
320 GOSUB 600
330 IF X$=CHR$(133) THEN EI$="":MA=0:RETURN
340 IF Y=0 THEN 300
350 IF MA=0 THEN TI$="000000":MA=1
360 X=VAL(MID$(TI$,5,2))
370 IF X>5 THEN MA=0:EI$="":Z2=0:TI$="000000"
380 IF X>5 THEN I=1:GOSUB600:GOSUB700:IF TI$="000059" THEN 300
385 IF X>5 THEN IF R2$="R" THEN R2$="":GOTO 330
390 IF X>5 THEN 380
395 IF Y=64 THEN ZI$=STR$(6):GOTO480
400 IF Y=32 THEN ZI$=STR$(5):GOTO480
410 IF Y=16 THEN ZI$=STR$(4):GOTO480
420 IF Y=8 THEN ZI$=STR$(3):GOTO480
430 IF Y=4 THEN ZI$=STR$(2):GOTO480
440 IF Y=2 THEN ZI$=STR$(1):GOTO480
450 IF Y=1 THEN ZI$=STR$(0):GOTO480
460 IF Y=65 THEN ZI$=STR$(7):GOTO480
470 IF Y=66 THEN ZI$=STR$(8):GOTO480
475 IF Y=67 THEN ZI$=STR$(9):GOTO480
480 EI$=EI$+ZI$
490 Y=PEEK(PO+1)AND127:IF Y=0 THEN530
500 IF R2$<>"W" THENZ2=VAL(MID$(TI$,5,2)):R2$="W"
510 IF VAL(MID$(TI$,5,2))-Z2>1 THEN R2$="":X=6:GOTO 370
520 GOSUB 700:GOTO 490
530 Z2=0:R2$=""
540 IF C$=EI$ THEN POKEPO+1,PEEK(PO+1)OR 128 :FOR K=1TO50:GOSUB700:NEXT
550 IF C$=EI$ THEN POKE PO+1,PEEK(PO+1)AND127:X=6:GOTO 370
560 IF LEN(EI$)>=5 THEN X=6:GOTO 370
570 GOTO 300
600 REM FINE PROGRAMMA *****************
610 GETX$:IF X$=CHR$(133)THEN R2$="R"
620 RETURN
700 REM PROTOCOLLO *********************
710 PRINTCHR$(19)
720 IFEI$=""THENPRINT"INPUT: ";CHR$(158);"NESSUN INPUT!";CHR$(144):GOTO 735
730 PRINT"INPUT: ";CHR$(158);EI$;CHR$(144);"                                    "
735 IF C$="" THEN C$="NON INTRODOTTO UN CODICE!!"
740 PRINT:PRINT"CODICE = ";CHR$(158);C$;CHR$(144):PRINT
750 PRINT"ATTESA:";CHR$(158);VAL(MID$(TI$,5,2));CHR$(157);CHR$(144);"    "
760 IF I>0 THEN Z$="NO":GOTO 780
770 Z$="SI"
780 PRINT:PRINT"PORTINAIO PRESENTE ? ";CHR$(158);Z$;CHR$(144);"          "
790 IF (PEEK(PO+1)AND128)=128 THEN Z$="ACCESO":GOTO 800
795 Z$="SPENTO"
800 PRINT:PRINT"DISPOSITIVO DI APERTURA DELLA PORTA: ";CHR$(158);Z$;
CHR$(144);"   "
810 RETURN
```

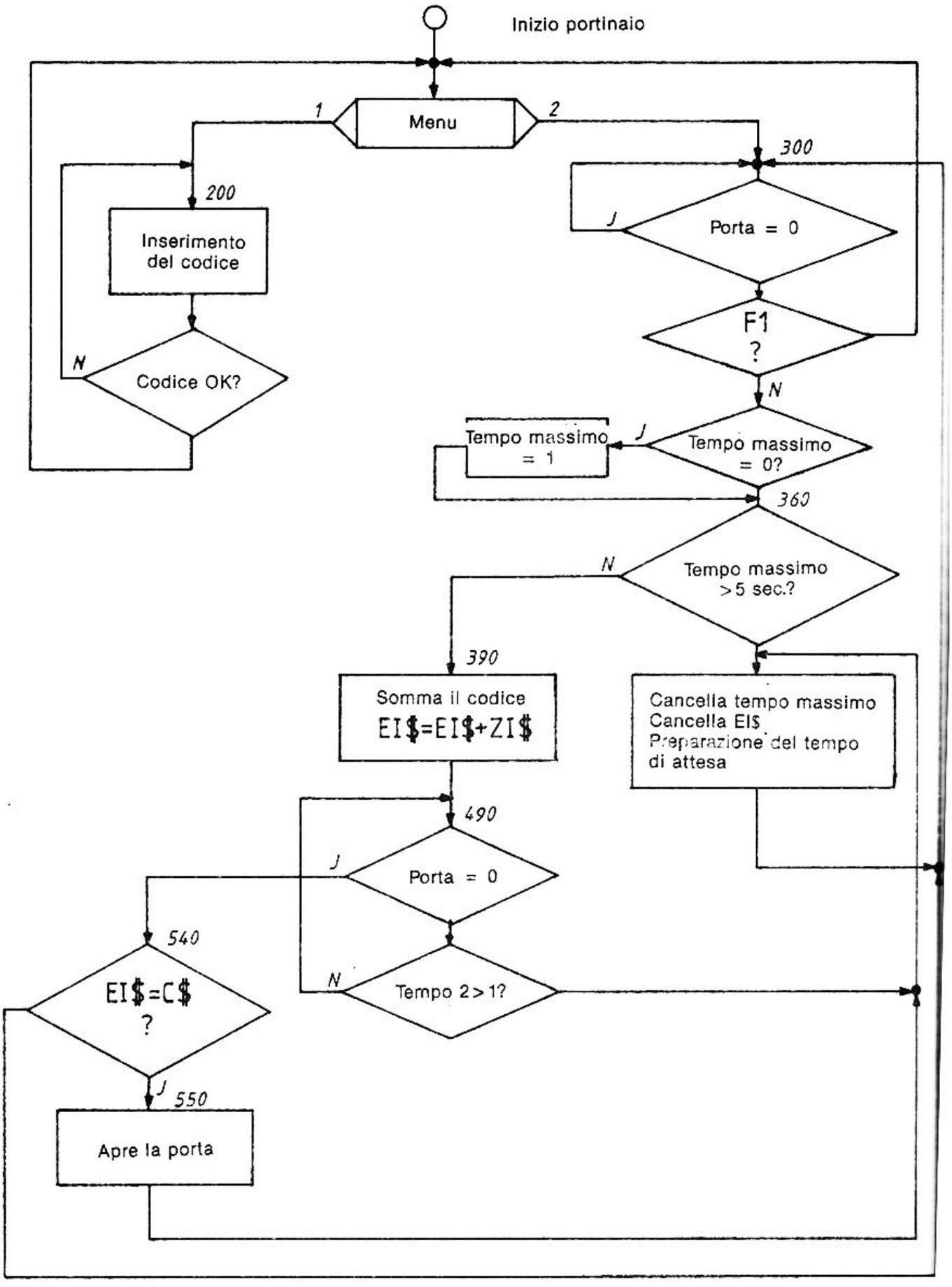

Inizio portinaio
Menu
1
2
200
Inserimento del codice
N
Codice OK?
300
J
Porta = 0
F1 ?
N
J
Tempo massimo = 1
Tempo massimo = 0?
360
N
Tempo massimo >5 sec.?
390
Somma il codice
EI$=EI$+ZI$
Cancella tempo massimo
Cancella EI$
Preparazione del tempo di attesa
490
J
Porta = 0
N
Tempo 2>1?
540
EI$=C$?
J
550
Apre la porta

Fig. 4

Il C64 come pilota

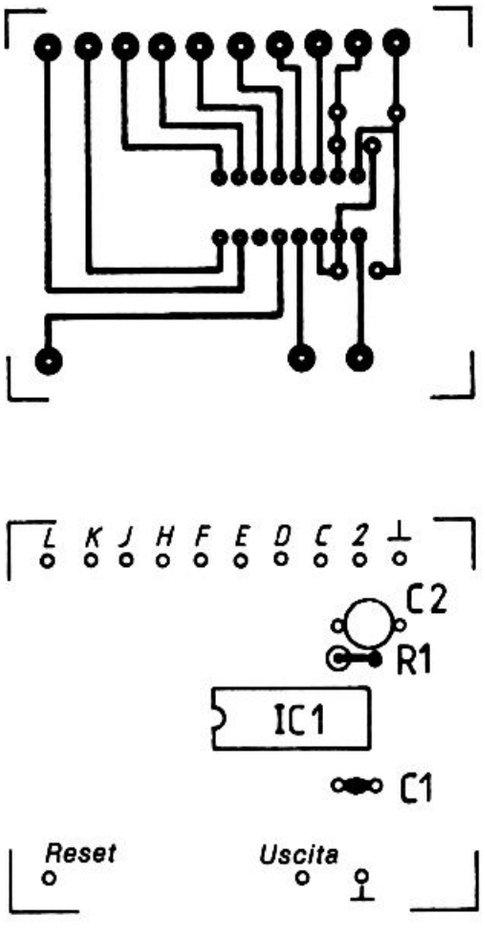
Il circuito descritto ed il relativo programma non devono essere usati soltanto come parte digitale di un convertitore analogico-digitale, bensì potrebbero ad esempio essere impiegati per un generatore di tensione regolabile. L'amplificatore di segnale seguente potrà selezionare il corrispondente fattore di amplificazione attraverso un commutatore. In questo modo è possibile generare una tensione regolabile dal calcolatore che può essere ad esempio utilizzata per comandare un motore e quindi regolarne il numero di giri.
I terminali di ingresso a disposizione sulla User-port PB7-PB0 possono rappresentare come segnale digitale un numero compreso tra 0 e 255. Collegando ad essi un convertitore digitale-analogico, verrebbe generata un tensione analogica variabile tra 0 e 2.5 Volt in corrispondenza col segnale di ingresso digitale. Con l'amplificatore di segnale qui descritto, si potrebbe amplificare questa uscita da 0 a 5 Volt in modo da raggiungere il livello TTL. Per l'elaborazione di questi segnali con circuiti TTL ciò non è comunque importante, dato che un segnale TTL è riconosciuto come segnale alto al di sopra di 1,7 Volt.

Il circuito

La realizzazione del circuito è relativamente semplice dato che esso è costituito di pochi componenti. Nucleo del circuito è naturalmente il convertitore digitale-analogico. A questo circuito integrato vanno collegati due condensatori ed una resistenza dopo di che il nostro convertitore digitale-analogico è pronto. Il terminale di reset del circuito integrato è collegato al positivo dell'alimentazione e perciò l'integrato è costantemente predisposto per ricevere i segnali della User-port. In uscita si ha a disposizione un segnale analogico compreso tra 0 e 2.5 Volt che dipende dallo stato degli ingressi PB0-PB7.

Il programma

Per prima cosa nella linea 20 viene assegnato alla variabile PO l'indirizzo fondamentale della porta 56576. Si cancella quindi lo schermo e, sulla terza riga, viene scritto il messaggio F3=Fine del programma. Nella linea 60 il registro 650 viene posto uguale a 128 con POKE, perciò con la pressione del tasto funzione viene continuamente accettato un valore dalla tastiera. Questa è la cosiddetta funzione di autoripetizione. Se il registro 650 è azzerato, la funzione di autoripetizione viene nuovamente annullata. Nelle linee 70-80 il registro di direzione dati PO+3 viene assegnato in uscita e le uscite vengono azzerate. Nelle righe successive dalla 90 alla 140, si trova l'effettivo nucleo del programma per la regolazione di un motore, ovvero per la regolazione della tensione analogica. Nella riga 100 viene dapprima assegnato alla variabile X$ un numero da tastiera. Se il tasto premuto è il



Convertitore digitale-analogico

Il C64 come pilota

tasto funzione F1=CHR$(133) e l'uscita PO+1 è maggiore di 0, allora al contenuto del registro PO+1 viene sottratto 1 e il risultato viene conservato con POKE nello stesso registro. Si salta poi alla linea 90.
Se viene premuto il tasto funzione F7 ed il contenuto del registro PO+1 è inferiore a 255, allora al contenuto di questo registro viene sommato un 1 e il risultato viene nuovamente conservato nello stesso registro PO+1. Si salta poi alla linea 90.
I tasti funzione F1 e F7 sono quindi correlati con l'aumento e la diminuzione del valore presente sulla porta. Se si preme il tasto F3, nella linea 130 si pone la porta a zero ed il programma ha termine, altrimenti con la linea 140 si salta nuovamente alla linea 90.

Elenco dei componenti del convertitore digitale-analogico

R1 = 390 Ohm
C1 = Condensatore al tantalio 10 microF
C2 = Condensatore al tantalio 100 microF
IC1 = ZN428.

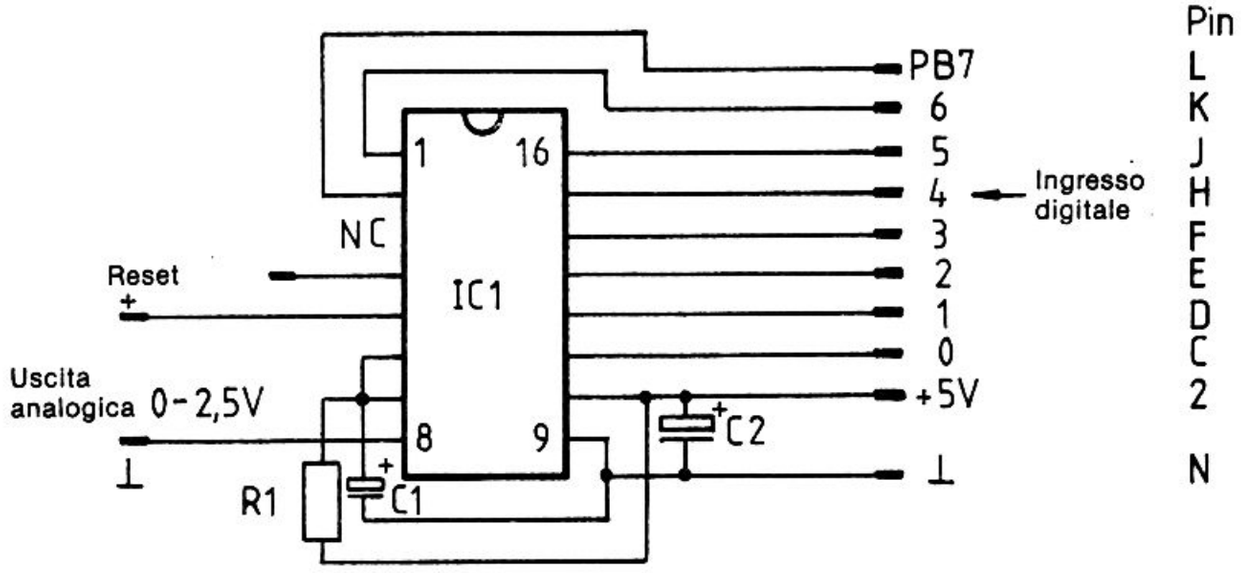


Fig. 2

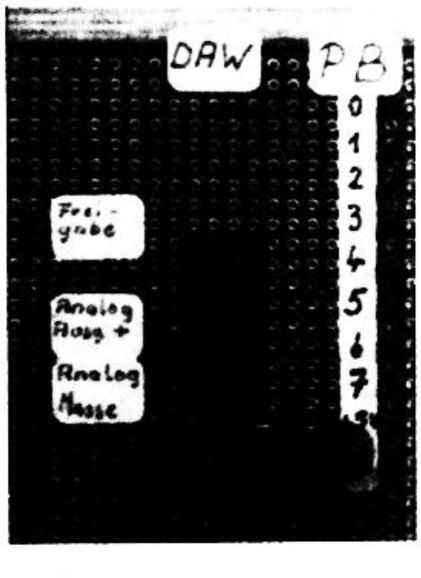


Fig. 3

```
10 REM C64 COME PILOTA
20 PO=56576
30 PRINTCHR$(147)
40 PRINT
50 PRINT"TASTO F3 = FINE PROGRAMMA"
60 POKE650,128:REM RIPETIZIONE
70 POKEPO+3,255:REM TUTTE LE PORTE IN USCITA
80 POKEPO+1,0:REM TUTTE LE USCITE A 0
90 PRINTCHR$(19);PEEK(PO+1);CHR$(157);"  "
100 GETX$:IFX$=""THEN100
110 IFX$=CHR$(133)ANDPEEK(PO+1)>0THENPOKEPO+1,
PEEK(PO+1)-1:GOTO90
120 IF X$=CHR$(136)ANDPEEK(PO+1)<255THENPOKEPO
+1,PEEK(PO+1)+1:GOTO90
130 IF X$=CHR$(134)THENPOKEPO+1,0:END
140 GOTO90
```

Il C64 come angelo custode

La sorveglianza di una stanza con un allarme è presentata nel listato seguente. Questo programma è in grado di sorvegliare due stanze e genera all'ingresso in una stanza un segnale di uscita sulla Userport. Vi è quindi un'uscita per ogni stanza che rimane inserita fintantoché la stanza corrispondente non indica il conteggio 0. Per esempio si può accendere la luce all'ingresso in una stanza e spegnerla nuovamente quando tutte le persone hanno lasciato la stanza.
In questo caso può anche essere inserito un segnale di allarme che si spegne non appena la stanza viene abbandonata. Si può anche pensare ad un accumulatore esterno che permette l'accensione di una sirena che suona finché, attraverso la pressione manuale di un tasto, l'allarme non viene disinserito.

Il programma

Nelle linee da 40 a 80 vengono presentate sullo schermo le immagini delle stanze. Il numero della stanza da sorvegliare viene assegnato alla variabile RA dopo di che segue, nella linea 80, un comando GOSUB alla routine 600. Il controllo della stanza avviene nella stessa maniera descritta in "IL C64 COME GUARDIANO". Occorre soltanto cancellare nella linea 720 la chiamata alle subroutine 750 e 800; lo stesso vale per la linea 840 nella quale vengono cancellati i GOSUB 950 e 1000. Nella linea 650 viene azzerata l'uscita attiva per la stanza 1, ovvero PB4, quando il contatore Z1 per la stanza 1 è inferiore a 1. Nella linea 685 viene viceversa portata a 1 l'uscita quando il contatore della stanza è maggiore o uguale a 1 (Z1 > =1). La stessa cosa vale nelle linee 700 e 735.
Se ora deve essere installato un segnale di allarme che può essere collegato con un circuito esterno, si può semplicemente cancellare le linee 700 e 650. Il segnale di allarme viene inserito mediante le linee 685 e 735 rispettivamente e provoca il suono della sirena. Un'interruzione attraverso il programma non è così più possibile, anche quando il numero di persone presenti nella stanza è uguale a 0. Per interrompere il segnale di allarme occorre interrompere il programma.
Va ancora una volta ricordato che con un tale dispositivo d'allarme, l'inserimento del dispositivo può essere fatta contemporaneamente alla memorizzazione dell'ora in una variabile. Dopo questa memorizzazione, l'ora viene indicata sullo schermo o, ancora meglio, viene scritta su una stampante.
Si può quindi portare il calcolatore in un ciclo senza fine e quindi bloccare stabilmente a 1 anche l'uscita.

```
5 REM C64 COME ANGELO CUSTODE
10 POKE53248+32,6:POKE53248+33,6
20 M$="■ ********* CONTROLLO STANZE *********":PRINTM$
30 PRINT:PRINT
40 INPUT"MASSIMO 2 LOCALI:";RA
50 IF RA>2ORRA<=0THENEND
60 FORI=1TORA:INPUT"NOME LOCALI MASSIMO 10 LETTERE:";R$(I):NEXT
70 PRINTM$
80 ON RA GOSUB 400,300:GOSUB600
300 PRINT"■■■ ■";:FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
310 FORK=1TO8:PRINT" ■ ■                    ■ ■                    ■ ■":NEXT
320 PRINT" ■";
330 FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
340 X=4:Y=5:GOSUB5000:PRINTR$(1)
350 X=22:Y=5:GOSUB5000:PRINTR$(2)
360 RETURN
400 PRINT"■■■ ■";:FORK=2TO20:PRINT" ";:NEXT:PRINT"■"
410 FORK=1TO8:PRINT" ■ ■                    ■ ■":NEXT
420 PRINT" ■";
430 FORK=2TO20:PRINT" ";:NEXT:PRINT"■"
440 X=4:Y=5:GOSUB5000:PRINTR$(1)
450 RETURN
600 REM INIZIO DEL CONTROLLO
```

```
605 PO=56576:POKEPO+3,240:POKEPO+1,PEEK(PO+1)ANDNOT240
610 PRINT"■":X=1:Y=24:GOSUB5000:PRINT"F1 = FINE DEL PROGRAMMA";
620 FORI=1TORA:ONIGOSUB 650,700:NEXT:GOTO840
630 GETX$:IFX$<>"■"THEN620
640 PRINT"■■":END
650 IFZ1<1THENPOKEPO+1,PEEK(PO+1)ANDNOT16
655 P=PEEK(PO+1)AND3:IFP=2THENM1=1:RETURN
660 IFM1THENIFP=0THENM1=0:RETURN
665 IFM1=0THEN:RETURN
670 IFP=1THENZ1=Z1+1:M1=0
675 IFP<>1THENRETURN
680 X=3:Y=7:GOSUB5000:PRINTZ1
685 IFZ1>=1THENPOKEPO+1,PEEK(PO+1)OR16
690 RETURN
700 IFZ2<1THENPOKEPO+1,PEEK(PO+1)ANDNOT32
705 P=PEEK(PO+1)AND12:IFP=8THENM2=1:RETURN
710 IFM2THENIFP=0THENM2=0:RETURN
715 IFM2=0THENRETURN
720 IFP=4THENZ2=Z2+1:M2=0
725 IFP<>4THENRETURN
730 X=21:Y=7:GOSUB5000:PRINTZ2
735 IFZ2>=1THENPOKEPO+1,PEEK(PO+1)OR32
740 RETURN
840 FORK=1TORA:ONKGOSUB850,900
845 NEXT:GOTO630
850 IFZ1>0THENP=PEEK(PO+1)AND3:IFP=1THENM5=1:RETURN
860 IFM5THENIFP=0THENM5=0:RETURN
865 IFM5=0THEN:RETURN
870 IFP=2THENZ1=Z1-1:M5=0
875 IFP<>2THENRETURN
880 X=3:Y=7:GOSUB5000:PRINTZ1;"■  ":RETURN
900 IFZ2>0THENP=PEEK(PO+1)AND12:IFP=4THENM6=1:RETURN
910 IFM6THENIFP=0THENM6=0:RETURN
915 IFM6=0THENRETURN
920 IFP=8THENZ2=Z2-1:M6=0
925 IFP<>8THENRETURN
930 X=21:Y=7:GOSUB5000:PRINTZ2;"■  ":RETURN
5000 REM POSIZIONAMENTO DEL CURSORE
5005 MS=40:MZ=25
5010 IFX>MSORY>MZORY-2<0ORX-1<0THENRETURN
5015 POKE214,Y-2:PRINT
5020 POKE211,X-1
5025 RETURN
```

Il C64 come giardiniere

Non tutti gli appassionati di floricoltura possiedono un calcolatore e quindi devono cercare un aiuto per innaffiare i fiori durante le loro ferie. Voi, come possessori di un calcolatore, potete utilizzarlo come innaffiatore per i vostri fiori. Si pone ovviamente la domanda in base a quale principio vadano innaffiati i fiori. Fondamentalmente, i fiori sui balconi possono essere bagnati ad intervalli di tempo periodici. All'interno dei vasi occorre prevedere un sistema automatico per l'eliminazione dell'ac-

qua in eccesso in quanto i fiori non possono vivere immersi. Ogni appassionato di botanica avrebbe sicuramente i capelli dritti dall'orrore! I fiori non possono annegare! Inoltre i fiori richiedono modi diversi di innaffiamento. Il modo più semplice sarebbe quello di innaffiare i fiori secondo l'umidità della terra. Con due chiodini ed un piccolo circuito elettronico è possibile realizzare con un calcolatore un buon dispositivo per l'innaffiamento automatico.

Il circuito

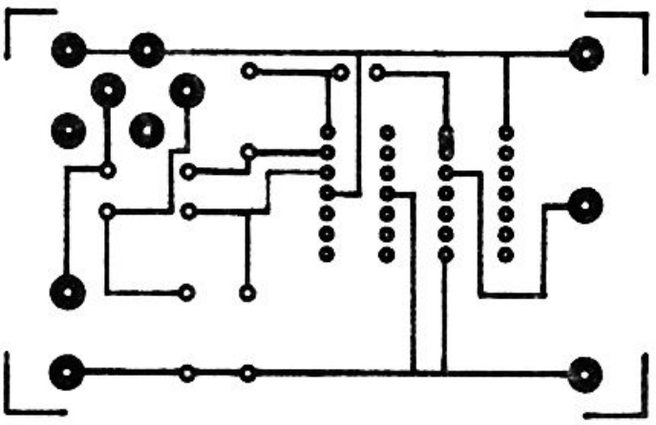
La terra dei fiori bagnata o umida ha una conducibilità maggiore della terra asciutta. Due chiodini vengono usati come sensori e piantati nella terra. In questo modo sull'ingresso 2 dell'operazionale vi è una piccola differenza di potenziale che può essere regolata per mezzo del potenziometro R1. Con il secondo partitore di tensione R2 e R3, viene generata sull'ingresso 3 una tensione di riferimento che va regolata quando la terra è bagnata. Quando la terra si è nuovamente asciugata, la conducibilità diminuisce, ovvero la resistenza aumenta, così come la tensione sull'ingresso 2. Quando questa tensione di comando è positiva rispetto alla tensione di riferimento allora l'uscita dell'amplificatore operazionale diventa negativa e si ha la commutazione del trigger di Schmitt. Sull'ingresso della porta vi è a disposizione uno stato logico 1 (+5 Volt).
La regolazione del livello di aridità consentito per la terra può essere quindi effettuata per mezzo della tensione di riferimento e può essere determinata per mezzo dei potenziometri R3 o R1.

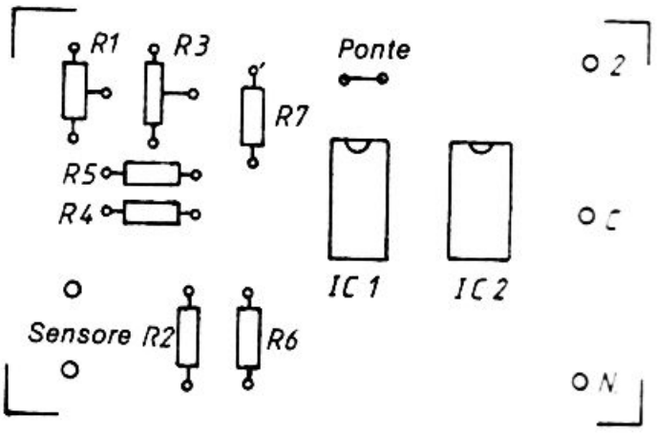
Il programma

Per innaffiare i fiori è necessaria una pompa per l'acqua che possa essere alimentata a 220 Volt. Il vostro C64 non è in grado di commutare una tensione così alta e sarà quindi necessario usare un cosiddetto interruttore di potenza. Per questa realizzazione e per le funzioni collegate consultate il capitolo sulle "Prese di corrente programmabili". Come ingresso di misura del circuito elettronico, è stato scelto l'ingresso PB0. Il terminale PB7 serve invece come uscita per l'interruttore di potenza. Il programma esemplificativo che segue è predisposto per l'aquisizione di un solo valore da misurare. Se è necessaria l'acquisizione di più valori, allora occorre assegnare la variabile fissa per mezzo di una variabile temporanea. Ciò significa che l'intero programma viene riciclato mediante una istruzione FOR ed i parametri esistenti d'ingresso e di uscita vengono assegnati alla variabile temporanea. L'accensione della pompa è provocata dalla misura di umidità dei sensori. Gli indirizzi fondamentali della porta sono indicati nel listato del programma all'interno delle linee di commento (REM).

Elenco dei componenti del dispositivo per l'innaffiamento automatico:

R1 = 250 KOhm
R2 = 1,5 KOhm
R3-R6 = 10 KOhm
R7 = 100 KOhm
IC1 = LM324
IC2 = SN74132
2 chiodini di acciaio

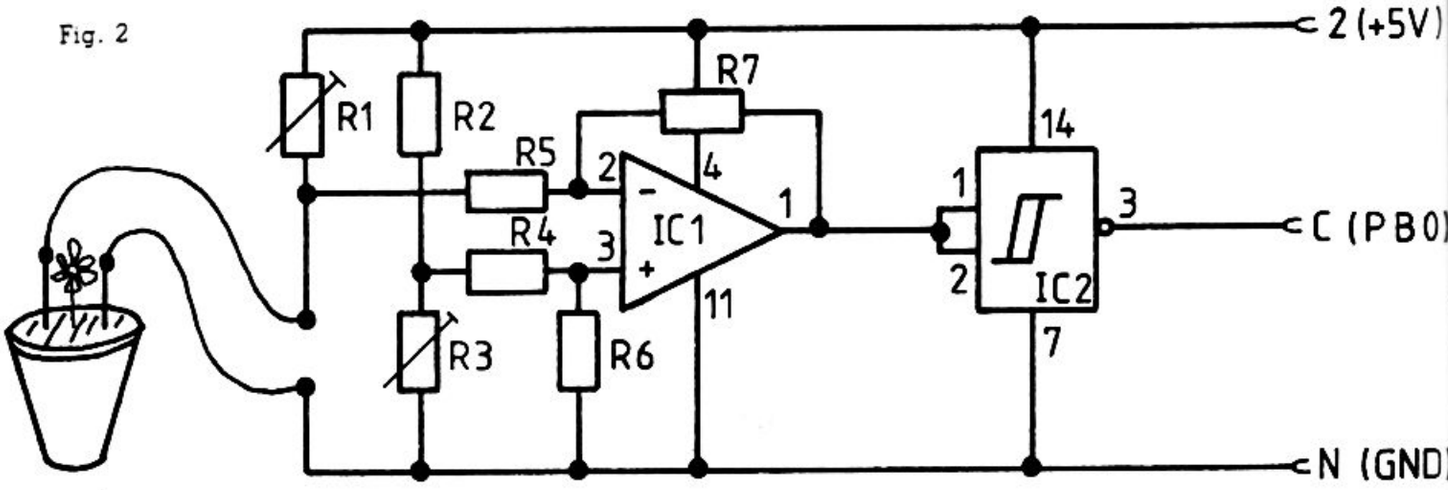




Il C64 come giardiniere

Fig. 1

Fig. 2




```
5 REM C64 COME GIARDINIERE
10 PRINTCHR$(147):PRINT"FINE PROGRAMMA = F1":PRINT
20 REM INDIRIZZO DI BASE DELLA PORTA B
30 PO=56576
40 REM REGISTRO=56576+3
50 POKE PO+3,128:REM PORTE 0-7= INGRESSO** PORTA 7 = USCITA
60 REM PORTA 7 = AZZERAMENTO
70 POKE PO+1,0
80 REM SE PORTA 0 = 1 ALLORA PORTA 7 = 1
90 IF(PEEK(PO+1)AND1)THENPOKEPO+1,128:FORK=1TO2000:NEXT
100 REM IST PORT 0 = 0 DANN PORT 7 = 0
110 IF(PEEK(PO+1)AND1)=0THENPOKEPO+1,0:FORK=1TO2000:NEXT
120 GETX$:IFX$=CHR$(133)THEN END
130 GOTO30
```

Il C64 come controllore ferroviario

Se, ad esempio, dei segnali ferroviari devono essere inviati al C64, questi devono in certe circostanze essere modificati. Se il segnale è troppo debole, deve essere amplificato, mentre se è troppo forte deve essere attenuato.

Queste modifiche del segnale possono essere ottenute mediante un amplificatore di segnali il quale è in grado di amplificarli fino ad un massimo di 5 Volt. Nel listato seguente, presentiamo un programma che legge un segnale d'ingresso sul terminale PB0 della User-port e lo mostra sullo schermo come un segnale su di un oscilloscopio. Con l'aiuto del circuito realizzato a questo scopo, potete amplificare il segnale fintantoché l'ingresso non sia visibile sullo schermo.

Descrizione del circuito

Il segnale d'ingresso giunge attraverso la resistenza R3 al terminale d'ingresso del circuito integrato. Per l'amplificatore servono le resistenze R1, che determina il fattore di amplificazione, e R2. Regolando il potenziometro R1 possiamo quindi modificare questo fattore. Il circuito integrato IC fornisce sull'uscita 1 un segnale amplificato di conseguenza. Questo segnale non può oltrepassare la tensione di alimentazione di 5 Volt. La massima tensione d'ingresso è invece di 24 Volt.

Il programma

Per leggere il segnale d'ingresso e per il suo trattamento come immagine grafica, è necessario un programma in linguaggio macchina. Un programma scritto in BASIC non può infatti fornire alcun risultato accettabile a causa del tempo impiegato dall'interpretazione delle istruzioni. Nel listato sono

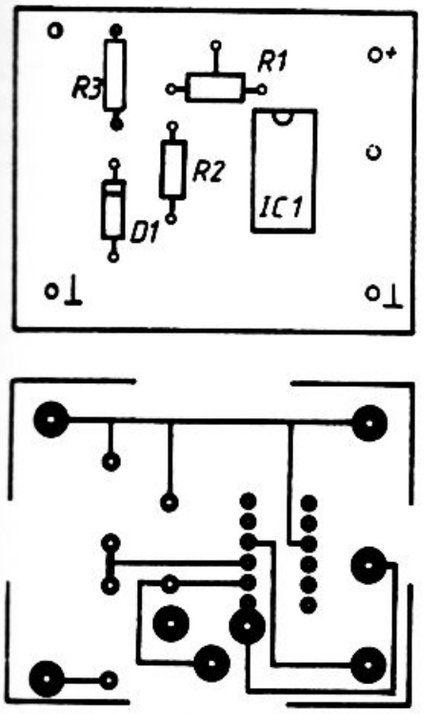


Fig. 1

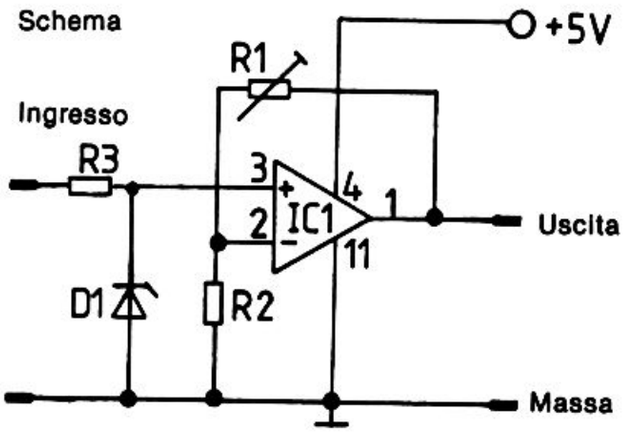


Fig. 2

riportati i valori da inserire con istruzioni POKE. Dopo l'inizio del programma con il comando RUN, compare in mezzo allo schermo un trattino bianco che simboleggia il segnale in ingresso sulla porta PB0. La pressione del tasto F3 provoca l'inizio di un processo di misura il cui risultato viene mostrato graficamente. Se desiderate una misura continua del segnale, premete quindi il tasto F3 ed in tal modo avrete un controllo costante del segnale. La pressione del tasto F1 provoca invece la fine del programma.

Elenco dei componenti dell'adattatore di segnali:

R1	=	10 KOhm
R2/R3	=	1 KOhm
D1	=	ZPD4,8
IC1	=	LM324

```
5 REM C64 COME CONTROLLORE
10 FOR I=49152 TO49603
20 READ X:POKEI,X:S=S+X
30 NEXT
40 IFS<>62858 THEN PRINT"ERRORE NEI DATI":END
50 SYS 49152
100 DATA 169,0, 133, 250, 141, 62, 3, 133
110 DATA 251,141, 3, 221, 169, 1, 141, 65
120 DATA 3,169, 128, 141, 138, 2, 32, 152
130 DATA 192,32, 135, 192, 169, 27, 133, 253
140 DATA 32,194, 192, 32, 152, 192, 169, 0
150 DATA 133,250, 141, 62, 3, 169, 100, 133
160 DATA 251, 173, 1, 221, 41, 1, 208, 7
170 DATA 169, 100, 133, 251, 76, 67, 192, 169
180 DATA 85, 133, 251, 32, 230, 192, 32, 146
190 DATA 193,230, 250, 208, 228, 169, 1, 141
200 DATA 62,3, 169, 0, 133, 250, 173, 1
210 DATA 221,41, 1, 208, 7, 169, 100, 133
220 DATA 251,76, 104, 192, 169, 85, 133, 251
230 DATA 32,230, 192, 32, 146, 193, 230,250
240 DATA 165,250, 201, 60, 208, 224, 32, 62
250 DATA 241,240, 251, 201, 134, 240, 164, 32
260 DATA 152,192, 32, 178, 192, 96, 96, 173
270 DATA 17,208, 9, 59, 141, 17, 208, 173
```

```
280 DATA 24,208, 9, 24, 141, 24, 208, 96
290 DATA 169,0, 133, 251, 169, 32, 133, 252
300 DATA 169,0, 170, 160, 0, 145, 251, 200
310 DATA 208,251, 232, 230, 252, 224, 32, 208
320 DATA 242,96, 169, 147, 32, 210, 255, 169
330 DATA 27,141, 17, 208, 169, 21, 141, 24
340 DATA 208,96, 169, 251, 133, 251, 169, 3
350 DATA 133,252, 165, 253, 162, 0, 160, 0
360 DATA 145,251, 200, 208, 251, 232, 230, 252
370 DATA 224,3, 208, 242, 160, 0, 145, 251
380 DATA 200,192, 237, 208, 249, 96, 169, 0
390 DATA 133,253, 133, 254, 165, 251, 133, 253
400 DATA 24,70, 253, 70, 253,70 ,253, 165,253
410 DATA  141, 60, 3, 165, 254, 141, 61
420 DATA  3, 24, 6, 253, 38, 254, 6
430 DATA  253, 38, 254, 6, 253, 38, 254
440 DATA  6, 253, 38, 254, 6, 253, 38
450 DATA  254, 6, 253, 38, 254, 6, 253
460 DATA  38, 254, 6, 253, 38, 254, 165
470 DATA  253, 141, 63, 3, 165, 254, 141
480 DATA  64, 3, 173, 60, 3, 133, 253
490 DATA  173, 61, 3, 133, 254, 24, 6
500 DATA  253, 38, 254, 6, 253, 38, 254
510 DATA  6, 253, 38, 254, 6, 253, 38
520 DATA  254, 6, 253, 38, 254, 6, 253
530 DATA  38, 254, 24, 173, 63, 3, 101
540 DATA  253, 133, 253, 169, 0, 101, 254
550 DATA  133, 254, 24, 173, 64, 3, 101
560 DATA  254, 133, 254, 165, 251, 41, 7
570 DATA  24, 101, 253, 133, 253, 169, 0
580 DATA  101, 254, 133, 254, 165, 250, 41
590 DATA  248, 24, 101, 253, 133, 253, 169
600 DATA  0, 101, 254, 133, 254, 24, 173
610 DATA  62, 3, 101, 254, 133, 254, 169
620 DATA  32, 101, 254, 133, 254, 96, 169
630 DATA  31, 229, 254, 48, 1, 96, 169
640 DATA  64, 229, 254, 16, 1, 96, 165
650 DATA  250, 41, 7, 170, 232, 169, 128
660 DATA  76, 172, 193, 74, 202, 208, 252
670 DATA  170, 173, 65, 3, 208, 6, 138
680 DATA  73, 255, 76, 188, 193, 138, 160
690 DATA  0, 17, 253, 145, 253, 96, 65
```

Il C64 come sorvegliante

Il programma seguente serve per la sorveglianza di un massimo di quattro stanze. La sorveglianza viene effettuata per mezzo di due barriere luminose che si trovano sulla porta d'ingresso nella stanza e sono disposte parallelamente. Quando una persona entra nella stanza, viene dapprima interrotta la barriera numero 1, dopo di che entrambe le barriere sono interrotte e, con l'ingresso nella

stanza, viene ad essere interrotta soltanto la barriera numero 2. Le due barriere risulteranno poi entrambe libere. Il programma è in condizione di riconoscere questo processo d'interruzione. Ad ogni stanza viene riservato un contatore: se una persona entra nella stanza il contatore viene incrementato di 1, quando la persona lascia la stanza il processo d'interruzione delle barriere luminose risulta invertito e il contatore viene diminuito di 1. In questa maniera il guardiano delle stanze può essere sempre a conoscenza del numero totale delle persone che si trovano nella stanza. È quindi un processo di conteggio. Che cosa si può fare con un circuito del genere?
Innanzitutto, si può effettuare una semplice sorveglianza delle stanze dato che sullo schermo vengono mostrati gli stati relativi alle stanze corrispondenti. In alternativa questo programma può essere utilizzato per l'accensione automatica delle luci ed il loro spegnimento non appena la stanza è nuovamente vuota. Attraverso il segnale di conteggio (numerico) si può anche inserire un dispositivo di allarme o un gioco per una festa come un motivetto musicale.

Il circuito

Per ogni stanza sono necessarie due barriere luminose a raggi infrarossi. Poiché la User-port può fornire in uscita un massimo di 100 mA circa, è necessario, per la realizzazione di questo circuito, un alimentatore esterno in grado di fornire +5 Volt. Le barriere luminose sono disposte parallelamente sulla porta in modo tale che, in seguito all'attraversamento di una persona, entrambe le barriere vengano interrotte in successione. La distanza tra queste barriere parallele va determinata in modo che, con l'attraversamento, si abbia sicuramente l'interruzione di almeno una barriera. La barriera luminosa è provvista di una resistenza R1 e fornisce, in condizioni di riposo, un segnale in ingresso al trigger di Schmitt invertente 74132. Poiché l'ingresso del trigger di Schmitt è 1, come uscita sulla User-port si avrà uno 0 finché la barriera luminosa non viene interrotta. In questo caso il fototransistor fornisce all'inverter un segnale nullo e l'inverter fornisce a sua volta un segnale 1 alla User-port, comunicando in questo modo che la barriera luminosa è stata interrotta. Questo processo funziona nello stesso modo per tutte le barriere ed è relativamente facile da capire. Per la costruzione del circuito occorre solamente tenere presente che l'alimentazione esterna della barriera luminosa non deve superare i 5 Volt. Se la barriera luminosa fosse collegata al terminale a +5 Volt della User-port, questa risulterebbe sovraccaricata.

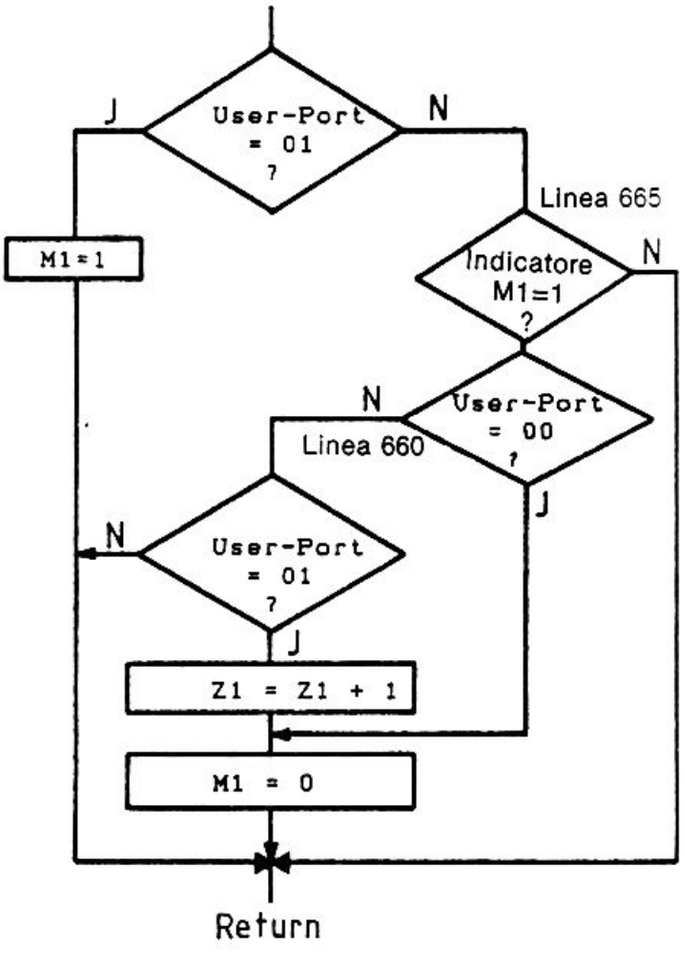


Fig. 1

Il programma

Dapprima vengono determinati i parametri d'ingresso mediante le linee dalla 5 alla 70. Ciò significa che viene fissato il numero delle stanze e la loro relativa denominazione. Relativamente alla variabile RA, viene poi mostrata sullo schermo l'immagine corrispondente. Ciò avviene nelle linee dalla 80 alla 450. Nella linea 80 si ha il comando GOSUB 600. Alla linea 600 comincia il programma per la sorveglianza delle stanze. Si inizia con l'assegnazione dell'indirizzo di base della porta (56576) alla variabile PO. Successivamente con POKE PO+3,0 la porta viene commutata come ingresso e con la linea 620 si inizia un ciclo FOR con la variabile contatore I. Ad ogni valore di I, si ha un comando GOSUB alle linee 650,700,750 e 800. Le singole subroutine si riferiscono alle misure degli ingressi delle singole stanze. Dopo il controllo di queste stanze, il programma viene inviato, con l'istruzione GOTO 840 della linea 620, alla routine per il controllo delle uscite. Qui viene nuovamente iniziato un ciclo con l'istruzione FOR e la variabile contatore K, e si ha, a seconda del valo-

re di K, il salto alle singole subroutine 850, 900, 950 e 1000. Queste subroutine provvedono al conteggio delle persone uscite dalle stanze. Dopo che sono state controllate le interruzioni delle barriere in seguito all'uscita delle persone, si ritorna alla linea 630 nella quale si controlla la pressione del tasto F1. Se questo tasto non è stato premuto, il programma salta alla linea 620 dove viene ripetuto l'intero processo di controllo degli ingressi e delle uscite.

Passiamo ora all'interruzione in ingresso e consideriamo ad esempio la subroutine 650. Per prima cosa si controlla se l'ingresso 2 è ancora a 1 mentre l'ingresso 1 è a 0. Questo si verifica quando la barriera luminosa 1 è interrotta mentre la barriera 2 è ancora libera. Si ha quindi il segnale 10 che per la User-port significa 2. Nella linea 650 viene quindi assegnato alla variabile M1 il valore 1 ed il programma ritorna all'inizio. Ora la persona prosegue ed interrompe entrambe le barriere luminose. Per la subroutine alla linea 650 ciò non ha alcun significato. Quando la persona entra nella stanza ed in tal modo interrompe la barriera luminosa 2 mentre la barriera 1 non è più interrotta, si verifica un evento. Questo evento è visibile nella linea 670 dato che in questo stato delle barriere la User-port ha valore 1. In questo caso il contatore Z1 della prima stanza viene incrementato di 1 e la variabile ausiliaria M1 viene cancellata. La variabile ausiliaria M1 è importante in quanto, ad esempio, una persona può interrompere la prima barriera e poi cambiare idea ed abbandonare la stanza. In questo caso entrambe le barriere hanno valore 0 e nella linea 660 la variabile M1 viene nuovamente azzerata.

Per l'interruzione delle barriere durante l'uscita dalle stanze, consideriamo ad esempio la subroutine 850. Qui viene dapprima controllato se il contatore numero 1 è maggiore di 0, dato che non può essere negativo. Se Z1 è maggiore di 0 allora viene controllato l'ingresso della porta. Uscendo dalla stanza, diventa attiva per prima la barriera 2 mentre la barriera 1 resta libera. In questo caso la User-port assume lo stato 1 e nella linea 850 viene posta a 1 la variabile ausiliaria M5. La persona interrompe ora anche la barriera 1 ed entrambe le barriere sono interrotte senza che ciò provochi alcun effetto sulla subroutine. Quando viene liberata la barriera 2, sulla User-port si ha lo stato 2 e nella linea 870 il contatore viene decrementato di 1. Inoltre la variabile ausiliaria viene azzerata nuovamente. La variabile M5 ha qui lo stesso scopo della variabile ausiliaria M1: se la barriera 2 viene interrotta brevemente, allora M5 viene azzerata nella linea 860 a causa dello stato nullo di entrambe le barriere. Con la codificazione in BASIC, il tempo impiegato per la sorveglianza di quattro stanze è appena sufficiente. Se occorre sorvegliare più stanze ciò è possibile soltanto con un circuito elettronico supplementare dato che la User-port dispone soltanto di 8 ingressi ed ogni controllo di stanza necessita di due ingressi. Lo stato dei contatori viene costantemente indicato sullo schermo. Questi contatori potrebbero essere inviati all'esterno mediante una parte della User-port purché sulla User-port ci siano ancora delle uscite libere. Se venissero sorvegliate solo due stanze, potrebbero quindi essere accese le luci nelle stanze corrispondenti finché qualche persona si trova in queste stanze. Quest'accensione dipende dalla User-port che funziona quindi in parte come ingresso ed in parte come uscita. La sezione di uscita può ad esempio essere collegata con le "Prese di corrente programmabili". Nel comando di elementi esterni occorre sempre considerare con quali tensioni si ha a che fare. É in ogni caso conveniente che il segnale di uscita della User-port sia elettricamente sconnesso dal circuito da comandare. Il procedimento di comando può essere effettuato mediante una singola istruzione. Si può ad esempio dire: IF CONTATORE 1 > 0 allora assegna una determinata uscita sulla porta 1. Per capire come è possibile assegnare una singola uscita o leggere un singolo ingresso, consultate il capitolo "Interfaccia".

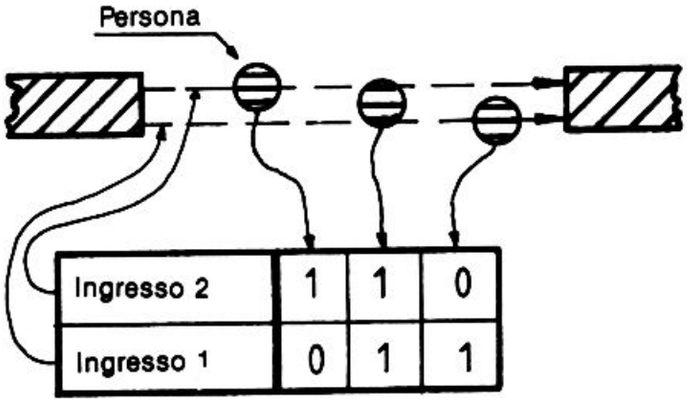


Fig. 2

Elenco dei componenti del guardiano

R1 = 2 x 40 Ohm 0,5 Watt
R2 = 2 x 4,7 KOhm
D1 = 2 x trasmettitore IR SFH 409
Tr1 = 2 x fototransistore IR SFH 309
IC1 = 1 x SN74132

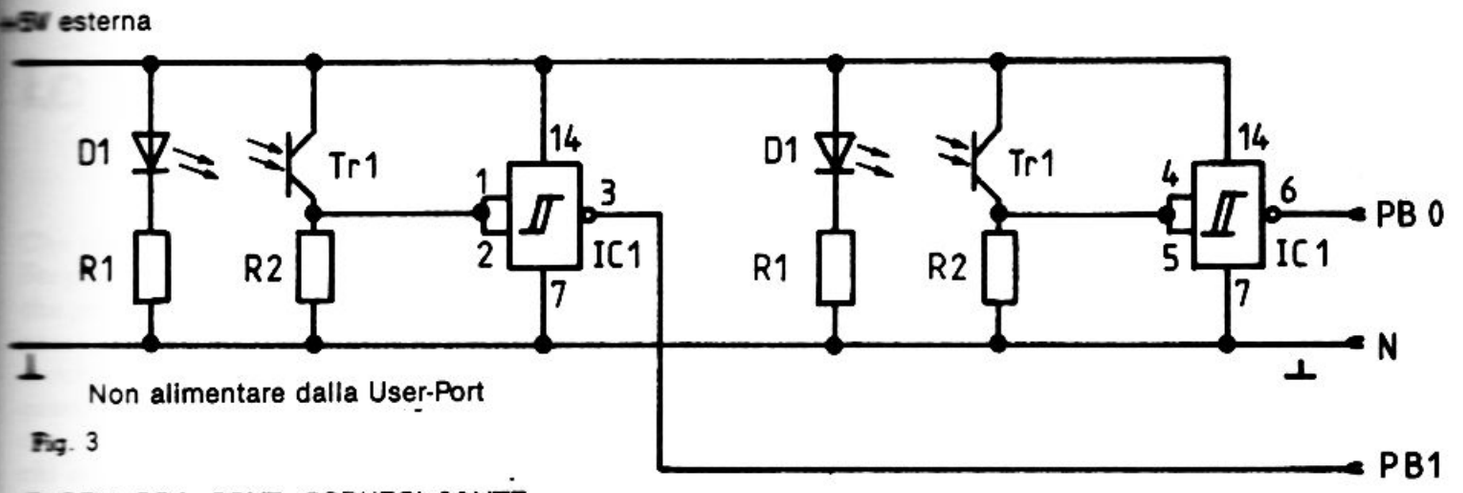


Fig. 3

```
5 REM C64 COME SORVEGLIANTE
10 POKE53248+32,6:POKE53248+33,6
20 M$="■ *** SORVEGLIANZA DI LOCALI ***":PRINTM$
30 PRINT:PRINT
40 INPUT"NUMERO MAX. LOCALI 4:";RA
50 IF RA>4ORRA<=0THENEND
60 FORI=1TORA:INPUT"NOME DEI LOCALI MAX.10 LETTERE:";R$(I):NEXT
70 PRINTM$
80 ON RA GOSUB 400,300,200,90 :GOSUB600
90 PRINT"■■■■ ■";:FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
100 FORK=1TO8:PRINT" ■ ■                    ■ ■                    ■ ■":NEXT
110 PRINT" ■";
120 FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
130 FORK=1TO8:PRINT" ■ ■                    ■ ■                    ■ ■":NEXT
140 PRINT" ■";
150 FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
160 X=4:Y=5:GOSUB5000:PRINTR$(1)
170 X=22:Y=5:GOSUB5000:PRINTR$(2)
180 X=4:Y=14:GOSUB5000:PRINTR$(3)
190 X=22:Y=14:GOSUB5000:PRINTR$(4)
195 RETURN
200 PRINT"■■■■ ■";:FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
210 FORK=1TO8:PRINT" ■ ■                    ■ ■                    ■ ■":NEXT
220 PRINT" ■";
230 FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
240 FORK=1TO8:PRINT" ■ ■                    ■ ■":NEXT
250 PRINT" ■";
260 FORK=2TO20:PRINT" ";:NEXT:PRINT"■"
270 X=4:Y=5:GOSUB5000:PRINTR$(1)
280 X=22:Y=5:GOSUB5000:PRINTR$(2)
290 X=4:Y=14:GOSUB5000:PRINTR$(3)
295 RETURN
300 PRINT"■■■■ ■";:FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
310 FORK=1TO8:PRINT" ■ ■                    ■ ■                    ■ ■":NEXT
320 PRINT" ■";
330 FORK=2TO38:PRINT" ";:NEXT:PRINT"■"
340 X=4:Y=5:GOSUB5000:PRINTR$(1)
350 X=22:Y=5:GOSUB5000:PRINTR$(2)
360 RETURN
400 PRINT"■■■■ ■";:FORK=2TO20:PRINT" ";:NEXT:PRINT"■"
410 FORK=1TO8:PRINT" ■ ■                    ■ ■":NEXT
420 PRINT" ■";
430 FORK=2TO20:PRINT" ";:NEXT:PRINT"■"
440 X=4:Y=5:GOSUB5000:PRINTR$(1)
450 RETURN
600 REM INIZIO SORVEGLIANZA
```

```
610 PO=56576:POKEPO+3,0:PRINT"■":X=1:Y=24:GOSUB5000:PRINT"F1 =
FINE DEL PROGRAMMA";
620 FORI=1TORA:ONIGOSUB 650,700,750,800:NEXT:GOTO840
630 GETX$:IFX$<>"■"THEN620
640 PRINT"■■":END
650 P=PEEK(PO+1)AND3:IFP=2THENM1=1:RETURN
660 IFM1THENIFP=0THENM1=0:RETURN
665 IFM1=0THEN:RETURN
670 IFP=1THENZ1=Z1+1:M1=0
675 IFP<>1THENRETURN
680 X=3:Y=7:GOSUB5000:PRINTZ1:RETURN
700 P=PEEK(PO+1)AND12:IFP=8THENM2=1:RETURN
710 IFM2THENIFP=0THENM2=0:RETURN
715 IFM2=0THENRETURN
720 IFP=4THENZ2=Z2+1:M2=0
725 IFP<>4THENRETURN
730 X=21:Y=7:GOSUB5000:PRINTZ2:RETURN
750 P=PEEK(PO+1)AND48:IFP=32THENM3=1:RETURN
760 IFM3THENIFP=0THENM3=0:RETURN
765 IFM3=0THENRETURN
770 IFP=16THENZ3=Z3+1:M3=0
775 IFP<>16THENRETURN
780 X=3:Y=16:GOSUB5000:PRINTZ3:RETURN
800 P=PEEK(PO+1)AND192:IFP=128THENM4=1:RETURN
810 IFM4THENIFP=0THENM4=0:RETURN
815 IFM4=0THENRETURN
820 IFP=64THENZ4=Z4+1:M4=0
825 IFP<>64THENRETURN
830 X=21:Y=16:GOSUB5000:PRINTZ4:RETURN
840 FORK=1TORA:ONKGOSUB850,900,950,1000
845 NEXT:GOTO630
850 IFZ1>0THENP=PEEK(PO+1)AND3:IFP=1THENM5=1:RETURN
860 IFM5THENIFP=0THENM5=0:RETURN
865 IFM5=0THEN:RETURN
870 IFP=2THENZ1=Z1-1:M5=0
875 IFP<>2THENRETURN
880 X=3:Y=7:GOSUB5000:PRINTZ1;"■■  ":RETURN
900 IFZ2>0THENP=PEEK(PO+1)AND12:IFP=4THENM6=1:RETURN
910 IFM6THENIFP=0THENM6=0:RETURN
915 IFM6=0THENRETURN
920 IFP=8THENZ2=Z2-1:M6=0
925 IFP<>8THENRETURN
930 X=21:Y=7:GOSUB5000:PRINTZ2;"■■  ":RETURN
950 IFZ3>0THENP=PEEK(PO+1)AND48:IFP=16THENM7=1:RETURN
960 IFM7THENIFP=0THENM7=0:RETURN
965 IFM7=0THENRETURN
970 IFP=32THENZ3=Z3-1:M7=0
975 IFP<>32THENRETURN
980 X=3:Y=16:GOSUB5000:PRINTZ3;"■■  ":RETURN
1000 IFZ4>0THENP=PEEK(PO+1)AND192:IFP=64THENM8=1:RETURN
1010 IFM8THENIFP=0THENM8=0:RETURN
1020 IFM8=0THENRETURN
1030 IFP=128THENZ4=Z4-1:M8=0
1040 IFP<>128THENRETURN
1050 X=21:Y=16:GOSUB5000:PRINTZ4;"■■  ":RETURN
5000 REM POSIZIONAMENTO CURSORE
5005 MS=40:MZ=25
5010 IFX>MSORY>MZORY-2<0ORX-1<0THENRETURN
5015 POKE214,Y-2:PRINT
5020 POKE211,X-1
5025 RETURN
```

Le prese di corrente programmabili

Che cos'è una presa di corrente programmabile? Per programmabile si intende una presa di corrente che possa essere comandata da un programma preparato da un programmatore. L'utilizzo dell'apparecchio collegato è perciò dipendente dal programma. In questo caso si tratta di una presa di corrente che può essere programmata anche da chi non conosce il BASIC.
Qui viene il bello! Il programma consiste di un generatore di programmi che permette, soltanto con pochi comandi, di sviluppare il programma per la presa di corrente. L'utente può, ad esempio, sviluppare il programma seguente mediante un menù di opzioni:

* PRESA 1=1
* PRESA 1=0
* TEMPO 1=20
* PRESA 3=Ingresso 1

Apriti Sesamo

Il programmatore ha la possibilità di alimentare la presa con il comando "Presa 1=1". Il comando "Presa 1=0" disinserisce invece nuovamente la presa di corrente 1. Come avete visto, è un uso facile e divertente. Inoltre esiste la possibilità di stabilire l'ora di accensione (temporizzatore). In questo modo la presa n. 1 può essere accesa e può poi partire un ciclo temporale con il comando "Tempo 1=20". In questo caso, trascorsi 20 secondi, il comando "Presa 1=0" sconnette nuovamente la presa. Ciò consentirebbe ad esempio di comandare un'apparecchiatura in modo completamente dipendente dal tempo. Poiché la presa di corrente utilizza tre uscite della User-port, rimangono ancora liberi cinque ingressi dei quali soltanto tre vengono utilizzati. La presa di corrente può essere anche alimentata attraverso un comando esterno. Il vostro garage, ad esempio, può essere aperto automaticamente lampeggiando i fari dell'automobile. Ecco il programma:

Presa 1=Ingresso 1

Questo semplice programma sta a significare che quando l'ingresso 1 è attivo, ovvero è allo stato 1, anche la presa 1 deve essere attiva. Se invece l'ingresso 1 è uguale a 0, allora anche la presa 1 è uguale a 0, cioè disinserita. Il programma può essere modificato secondo ogni desiderio del programmatore.

Il circuito

Il circuito integrato TDA1024 è un circuito rivelatore del passaggio per lo zero. L'alimentazione di rete a 220 Volt, 50 Hertz fornisce l'attraversamento dello zero che viene usato dal circuito rivelatore, il quale a sua volta fornisce sulla uscita 2 un impulso di trigger necessario per l'innesco del triac. I disturbi ad alta frequenza, altrimenti presenti, vengono in tal modo ridotti al minimo. Il TDA1024 viene alimentato direttamente a 220 Volt mediante la resistenza RV ed il condensatore CV. Il circuito rivelatore di zero, fornisce tutte le tensioni necessarie per l'alimentazione del circuito di trigger. Con la tensione costante presente sul pin 8, viene generata, attraverso i partitori di tensione R1-R2, una tensione di riferimento sul pin 4. Il comando d'ingresso vero e proprio del circuito integrato è il pin 5. In condizioni di riposo, quando dal calcolatore non giunge alcun impulso sul diodo emettitore del fotoaccoppiatore, il transistore fotosensibile ha una resistenza molto maggiore della resistenza R3. Perciò sull'ingresso di comando 4 vi è una tensione prossima al 0 Volt. Il circuito TDA1024 fornisce degli impulsi di comando al triac quando la tensione di comando è più alta della tensione di riferimento. Questo si verifica quando il diodo trasmettitore è attivato ed il transistore è perciò in conduzione cosicché sull'ingresso 5 vi è una tensione superiore a quella dell'ingresso 4. L'impulso di pilotaggio esce dal pin 2 del rivelatore del passaggio per lo 0 e, attraverso la resistenza RG, giunge sul gate del triac.

Da considerare attentamente

In linea di massima non dovrebbe essere possibile commettere alcun errore nella costruzione del circuito. Il fotoaccoppiatore serve per il trasferimento della tensione dal circuito al calcolatore. Chi permetterebbe volentieri che 220 Volt venissero inviati al suo calcolatore? La parte sinistra del circuito è alimentata a 220 Volt e perciò deve essere considerata con la massima attenzione in quanto tutti i pericoli sono qui radunati. Le tensioni superiori a 46 Volt possono essere mortali! Nella progettazione del circuito va considerato il carico sopportabile dal triac. Noi abbiamo usato un TIC226D che può sopportare una corrente di 6 Ampere con una tensione di lavoro di 400 Volt. Dipende comunque da voi, quale triac utilizzare. Fate quindi attenzione che il vostro circuito, particolarmente nella parte alimentata dalla rete, venga chiuso in una scatola di plastica in modo tale che non sia possi-

bile toccare in alcun modo la parte sotto tensione che è pericolosa. Se il circuito viene inserito all'interno della scatola che reca le prese di corrente da comandare, dovrebbe andare tutto per il meglio. Non dimenticatevi di collegare anche il connettore di massa giallo-verde. Ed ora buon divertimento con le prese di corrente programmabili.

Caratteristiche importanti del circuito:

* Elevato numero di commutazioni e lunga durata!
* Commutazione senza rimbalzi per l'assenza di parti in movimento
* Nessuna parte meccanica
* Nessuna perdita di contatto
* Compatibilità TTL
* Insensibilità agli urti

Il programma

Dopo il caricamento e l'avvio del programma con il comando RUN, compare sullo schermo un menù con tre possibilità di scelta:

1. Preparazione del programma
2. Avvio del programma
3. SAVE, LOAD, PRINT

Cominciamo con la preparazione del programma. Dopo la pressione del tasto 1, compare sullo schermo la riga M=MENÙ NUMERO MASSIMO DI LINEE=20 e più in basso il cursore lampeggiante più scuro. Il programma aspetta adesso l'inserimento dei comandi del programma. Ad ogni linea di comando, proprio come in BASIC, viene associato un numero di linea. Premete ora un tasto qualunque seguito dal tasto RETURN. Sullo schermo compare:

SYNTAX ERROR SONO PERMESSE AL MASSIMO LINEE DA 1 A 20. Più in basso si trova il cursore lampeggiante più scuro. Il calcolatore comprende naturalmente soltanto alcuni comandi ben determinati. Nell'elenco potete vedere tutti i comandi usati per la programmazione della presa di corrente.

```
PRESA 1=0
PRESA 2=0
PRESA 3=0
PRESA 1=1
PRESA 2=1
PRESA 3=1
PRESA 1=INGRESSO 1
PRESA 1=INGRESSO 2
PRESA 1=INGRESSO 3
PRESA 2=INGRESSO 1
PRESA 2=INGRESSO 2
PRESA 2=INGRESSO 3
PRESA 3=INGRESSO 1
PRESA 3=INGRESSO 2
PRESA 3=INGRESSO 3
END
LIST
TEMPO 1=XXX
TEMPO 2=YYY
TEMPO 3=ZZZ
GOTO
```

Consideriamo ora ogni singolo comando. Cominciamo con un facile circuito provocando il collegamento delle prese dalla 1 alla 3. Inserite le righe seguenti:

```
1 PRESA 1=1 >RETURN<
2 PRESA 2=1 >RETURN<
3 PRESA 3=1 >RETURN<
4 PRESA 1=0 >RETURN<
5 PRESA 2=0 >RETURN<
6 PRESA 3=0 >RETURN<
7 END
```

Questo piccolo programma inserisce una dopo l'altra le tre prese 1, 2 e 3 spegnendole poi nuovamente. Per dare inizio al programma, premiamo ora il tasto RETURN e sullo schermo comparirà nuovamente il menu. Mediante l'opzione 2 del menu, è possibile far partire il programma. Sullo schermo compare rapidamente la scritta:

"Passo del programma"

Sotto il passo del programma si trova una serie di numeri che rappresenta lo scorrere del programma e ci indica a quale passo è giunto il processore. Nel nostro caso la commutazione è troppo rapida ed il processore percorre rapidamente i passi da 1 a 7. Dopo di ciò il programma termina ed il processore ritorna al menù. Introduciamo ora una piccola modifica premendo perciò il tasto 1. Sullo schermo compare il cursore d'ingressso già noto e, dando il comando LIST >RETURN<, possiamo esaminare il nostro programma che compare sullo schermo. Modifichiamo la linea 7 in:

```
7 GOTO 1 >RETURN<
```

In tal modo la linea 7 precedente viene sostituita con un GOTO 1. Avete quindi visto come modificare le linee del programma. Indicate la linea di programma e premete poi il tasto RETURN per cancellare questa riga.
Il nostro programma è ora un programma senza fine poiché il processore trova nella linea 7 l'istruzione di salto alla linea 1. Inserite ora nuovamente M per tornare al menù ed iniziate il programma con il tasto 2. Sullo schermo compare nuovamente la scritta "Passo del programma" seguita dal nu-

mero corrispondente. In questo modo potete seguire l'avanzamento del programma. Siccome questo è un programma senza fine, per interromperlo dovete usare un tasto. Non usate per questo scopo il tasto STOP, altrimenti la condizione di uscita della User-port resta invariata. Per interrompere il programma usate invece il tasto F1. Premetelo adesso.

Comandi di tempo

I comandi temporizzati sono proprio ideali per le prese programmabili. Vi sono perciò dei comandi di temporizzazione. In questo programma vi sono a disposizione tre tempi che sono rispettivamente il tempo 1 in secondi, il tempo 2 in minuti ed il tempo 3 in ore. La precisione di questo temporizzatore non è proprio eccezionale ed è perciò consigliabile misurare il tempo durante lo svolgimento del programma, modificando poi eventualmente i parametri relativi. Un esempio: premete il tasto 1 e comparirà nuovamente il cursore di inserimento. Sostituiamo il vecchio programma con le seguenti istruzioni:

```
1 PRESA  1=1 >RETURN<
2 PRESA  2=1 >RETURN<
3 PRESA  3=1 >RETURN<
4 TEMPO 1=5 >RETURN<
5 PRESA  1=0 >RETURN<
6 TEMPO 1=5 >RETURN<
7 PRESA  2=0 >RETURN<
8 TEMPO 1=5 >RETURN<
9 PRESA  3=0 >RETURN<
10 END
```

Premendo poi il tasto M, ricompare il menu. Date inizio al programma con l'opzione 2 del menu stesso. Dopo poco tempo sullo schermo comparirà il numero 4. Trascorsi 5 secondi comparirà il numero 6. Dopo altri 5 secondi comparirà il numero 8 ed infine, ancora dopo 5 secondi, comparirà il numero 10 ed il programma avrà termine. Vedete quindi il funzionamento di un temporizzatore. La programmazione di un temporizzatore avviene assegnando alla variabile corrispondente, con una linea di programma, un valore che determina il tempo relativo. Il tempo da introdurre è limitato ad un massimo di tre caratteri e, come già accennato, i timer da 1 a 3 si riferiscono a tempi diversi. Provate con l'andamento del vostro programma il rispettivo funzionamento dei temporizzatori.

Floppy disk o cassetta

Il programma preparato deve essere memorizzato su un floppy disk o su una cassetta. Per fare ciò scegliete l'opzione 3 del menù e comparirà un menù secondario. Avete ora la possibilità di salvare il programma, di caricarlo o stamparlo. Cominciamo con la memorizzazione del programma. Inserite un dischetto inizializzato nel driver oppure una cassetta nel registratore e premete il tasto 1. Il programma richiede quindi il nome sotto cui salvare il programma. Potete dare un nome qualsiasi purché non venga superata la lunghezza massima di 16 caratteri e non vi siano dei caratteri particolari come:

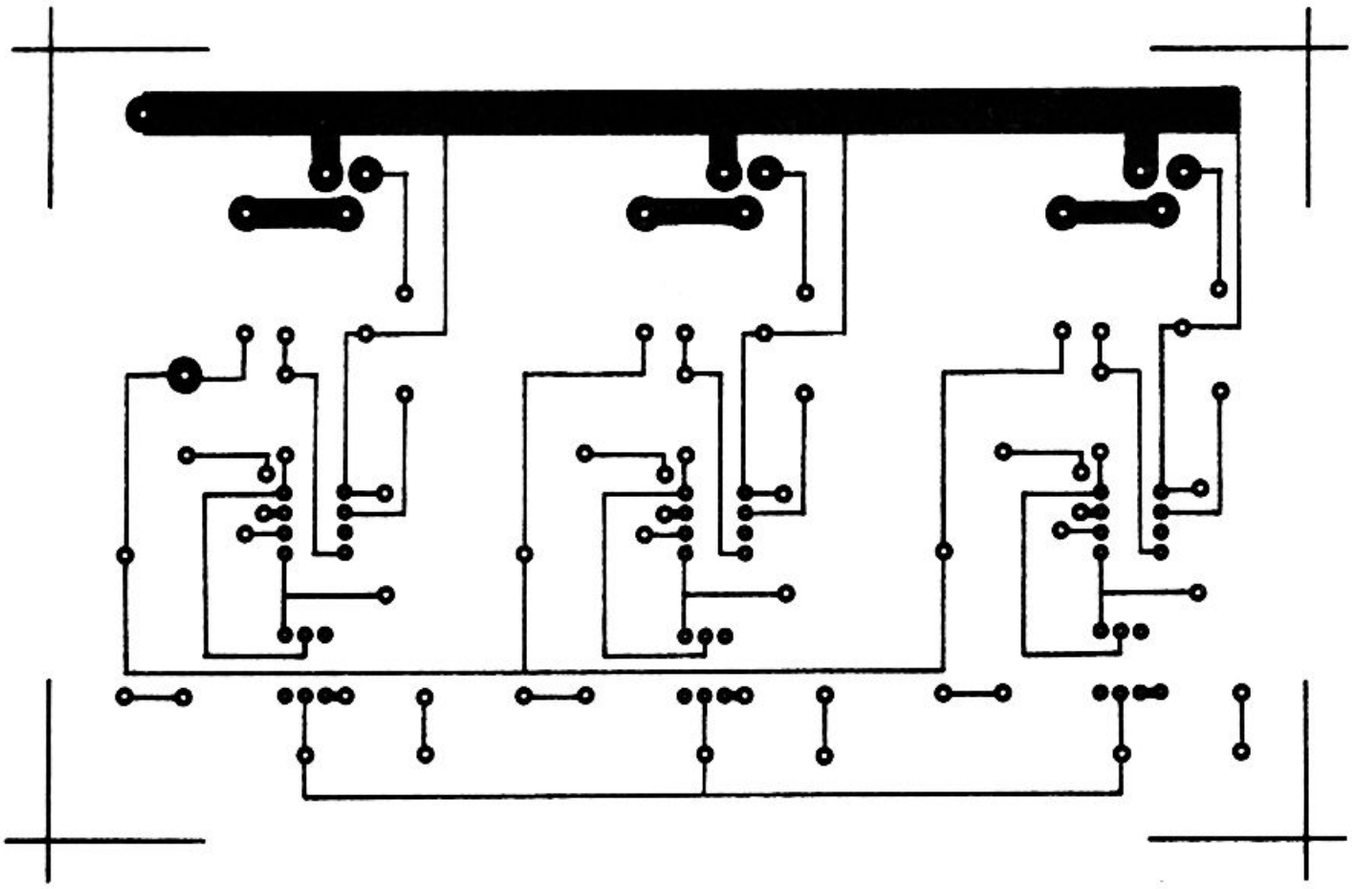
: ; , ? " $ % e così via

Date ora il nome relativo al vostro programma! Comparirà poi la domanda quale memoria di massa usare per la memorizzazione e potete scegliere tra dischetto e cassetta magnetica introducendo il numero corrispondente. Comincia quindi la memorizzazione del programma ed alla fine si torna al menù. Iniziate il programma con il tasto 2 e come potete vedere, il programma si trova ancora in memoria. Occorre ora controllare se il programma stesso è stato correttamente salvato sul dischetto o sul nastro. Per fare ciò, scegliete nuovamente l'opzione 1 del menù e listate il programma con il comando LIST. Cancellate ora, premendo 1 e poi RETURN, le linee da 1 a 10 e controllate con il comando LIST che il programma non sia più presente in memoria. Premete M e RETURN e tornate al menu. Scegliete adesso l'opzione 3 e ricomparirà il menù secondario già visto. Per caricare il programma scegliete l'opzione 2. Il programma vi chiederà ora il nome del programma da caricare e dopo il tasto RETURN viene nuovamente richiesto quale memoria di massa utilizzare per la lettura del programma memorizzato. Scegliete il numero corrispondente e vedrete che la memoria di massa inizia a funzionare ed a caricare il programma nella memoria di lavoro. Lanciate il programma con il tasto 2.

```
LISTATO DEL PROGRAMMA
PRESE PROGRAMMABILI

NOME DEL PROGRAMMA: TEST1
1 PRESA 1=1
2 PRESA 2=1
3 PRESA 3=1
4 TEMPO 1=5
5 PRESA 1=0
6 TEMPO 1=5
7 PRESA 2=0
8 TEMPO 1=5
9 PRESA 3=0
10 END

READY
```



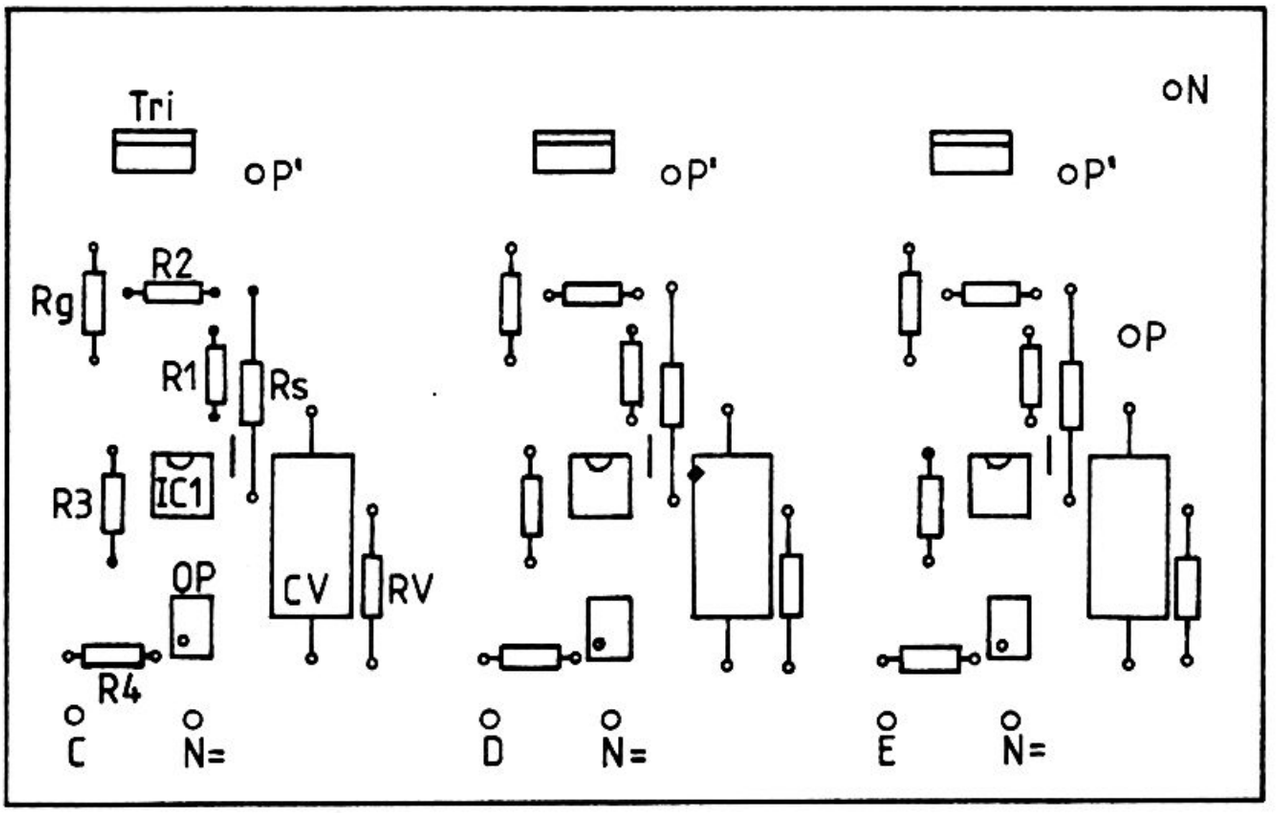


Fig. 1

Sicuramente è interessante poter documentare una determinata parte del programma. Per fare questo esiste il comando STAMPA. Scegliete dal menù principale l'opzione 3 e dopo la comparsa del menù secondario premete ancora il tasto 3. Segue la domanda se la stampante è OK o no. In caso affermativo rispondete "S". La vostra stampante inizierà a funzionare ed il programma verrà listato. Al listato è aggiunto anche il nome del programma. Ora però, non sono ancora state esaurientemente descritte le prese programmabili. Le prese utilizzano le uscite PB0-PB2 e perciò le porte PB3-PB7 sono ancora libere. Noi abbiamo programmato i canali PB3-PB5 come ingressi e possiamo, mediante questi ingressi, attivare direttamente le prese relative. Cosa significa ciò?

Apertura di un garage con i fari

Supponiamo che abbiate inserito il programma:

```
1 PRESA 1=INGRESSO 1 >RETURN<
2 GOTO 1 >RETURN<
```

Dopo l'avvio del programma con l'opzione 2, vedrete continuamente sullo schermo i passi 1 e 2. Fin qui tutto chiaro! Il processore assegna l'ingresso 1 alla presa 1, ovvero se l'ingresso è attivo, lo è anche la presa. Se invece l'ingresso 1 è a 0 allora la presa 1 non è collegata. In questo caso l'istruzione GOTO 1 è importante. Infatti il processore elabora i passi di programma così rapidamente che, in mancanza dell'istruzione GOTO, l'impulso corrispondente probabilmente non verrebbe preso dall'ingresso. Avete ora la possibilità di comandare le prese in funzione dello stato degli ingressi. In questo modo è stato raggiunto lo scopo. Ed ora un semplice esempio per aprire automaticamente la porta di un garage. Prima di tutto, sulla porta del garage si trova una fotocellula la quale, attraverso un circuito elettronico, reagisce ad una determinata serie di lampeggi luminosi. Se il lampeggio con i fari è corretto, allora il circuito elettronico invia un impulso sull'ingresso 1 della User-port (PB3). In questo modo è possibile aprire la porta del garage con l'istruzione PRESA 1=INGRESSO 1. Il segnale rimane invariato sul calcolatore fintantoché l'utente non lascia il garage e preme il tasto per la chiusura della porta. Attraverso la pressione del tasto viene interrotto anche il segnale sulla User-port. Per entrare nuovamente nel garage non è più possibile ora usare i fari. Per motivi di sicurezza il garage può essere aperto solamente con una chiave o ad esempio con un interruttore a chiave che provochi l'apertura della porta del garage. Quest'apertura è provocata anche dal solito segnale sulla User-port. Il programma sarà perciò semplicemente:

```
1 PRESA 1=INGRESSO 1 >RETURN<
2 GOTO 1 >RETURN<
```

Se la porta del garage raggiunge la sua posizione finale, il motore viene disinserito dal cosiddetto interruttore di fine corsa. In qualsiasi modo sia costruito, il comando esterno della porta di un garage può essere collegato al calcolatore mediante un singolo cavo.
Per concludere, ricordiamo che il programma è scritto in BASIC. Se più prese devono essere comandate contemporaneamente, allora il programma deve essere modificato. La modifica del programma per l'impiego con più prese è tuttavia conveniente soltanto se il programma è codificato in Assembler. Il funzionamento del programma attraverso l'interprete BASIC è relativamente lento cosicché il ritardo tra l'arrivo dell'impulso in ingresso e l'azione corrispondente può essere di disturbo. In casi normali e nell'uso casalingo si possono usare tre prese.

Scaccialadri

Con un temporizzatore, potrete programmare un circuito per spaventare i ladri. Potete con la presa 1 accendere la luce in cucina e spegnerla dopo un intervallo di due ore. Mediante la presa 2 accenderete quindi la luce in salotto contemporaneamente al televisore. Dopo un'altra ora la luce in salotto può essere spenta insieme al televisore. Per un tempo più breve potete ora accendere la luce in camera da letto simulando quindi il fatto di esservi coricati. L'intero ciclo può essere ripetuto con un tempo totale di 24 ore mediante una istruzione GOTO. In questo modo la vostra casa è praticamente abitata anche quando siete in ferie.

Elenco dei componenti delle prese programmabili:

IC1	=	TDA1024
Tri	=	Triac 226D
Rv	=	390 Ohm
Rs	=	180 Ohm
Rg	=	27 Ohm
Op	=	Fotoaccoppiatore CNY17
R1-R3	=	10 KOhm
R4	=	330 Ohm
Cv	=	0,22 microF 630 Volt

1 scatola
1 serie di prese da tavolo

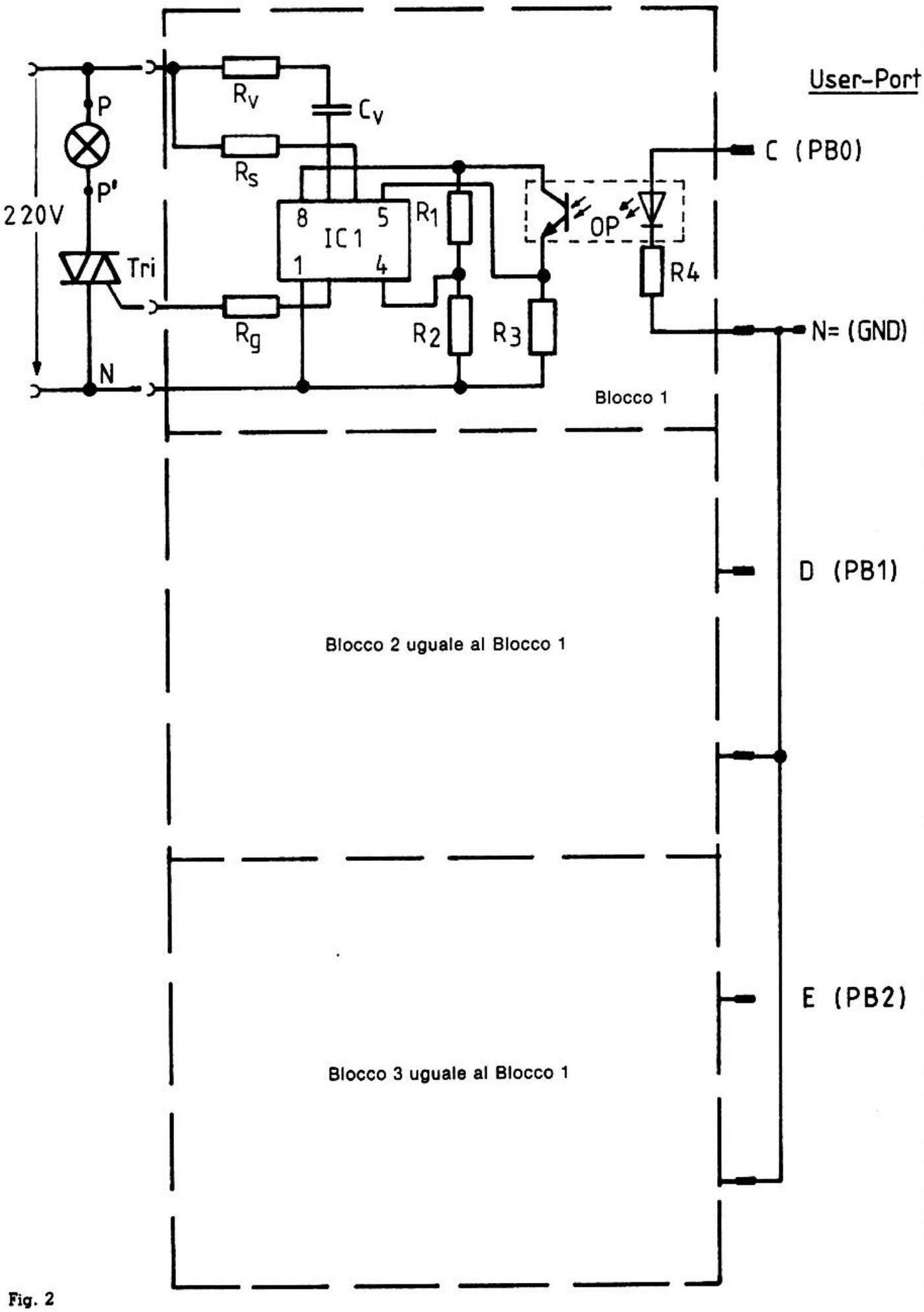

User-Port
C (PB0)
N= (GND)
D (PB1)
E (PB2)
220V
P
P'
N
Tri
Rv
Cv
Rs
Rg
IC1
8
5
1
4
R1
R2
R3
R4
OP
Blocco 1
Blocco 2 uguale al Blocco 1
Blocco 3 uguale al Blocco 1

Fig. 2

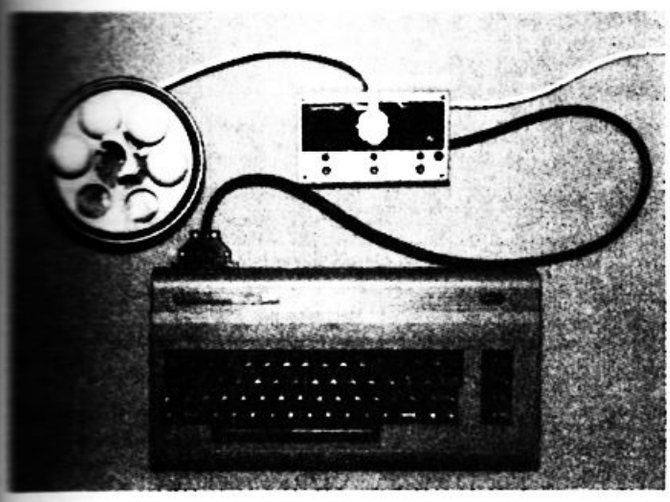


Fig. 3

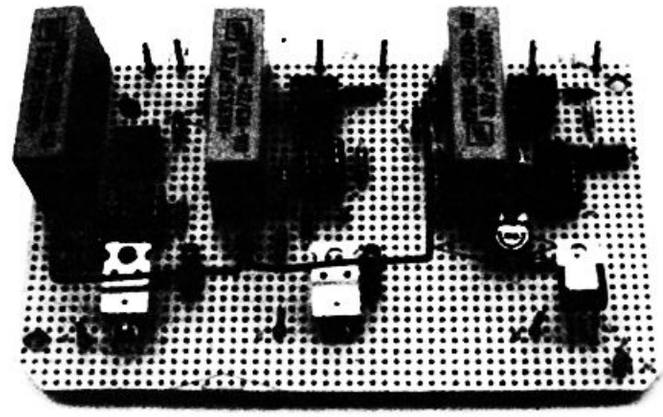


Fig. 4

```
1 REM PRESE DI CORRENTE PROGRAMMABILI
5 PO=56576:POKEPO+3,7:POKEPO+1,0:PRINT"■"
10 AB=16:VI=53248:POKE 650,128 :PRINT"■"
20 DIM ZN$(20),BC$(20),TE$(20),B$(AB)
30 FORI=1TOAB:READ B$(I):NEXT
40 GOTO 500
200 REM INIZIALIZZAZIONE DEL PROGRAMMA
205 M$="■M = MENU       NUMERO MAX. DI LINEE 20■":PRINTM$:PRINT
210 X=39 :GOSUB 5500
220 IF Z$="M" THEN RETURN
230 IF Z$="LIST"THEN GOSUB 4000:GOTO210
235 GOSUB300
240 GOTO 210
300 X$="":X1$="":R2$="":REM CONTROLLO LINEA
305 FORI=1TOLEN(Z$):IFMID$(Z$,I,1)=" "THEN 310
307 X$=X$+MID$(Z$,I,1)
310 NEXT:Z$=X$:IFZ$=""THENRETURN
312 IF LEN(Z$)>2THENX=2
313 IF LEN(Z$)<=2THENX=LEN(Z$)
315 ZN$="":FORI=1TOX
320 IF ASC(MID$(Z$,I,1))>=48AND ASC(MID$(Z$,I,1))<=57 THEN ZN$=ZN$+MID$(Z$,I,1)
325 NEXT
330 IFVAL(ZN$)<=0ORVAL(ZN$)>20THENPRINT"SYNTAX ERROR ":ER=1:GOSUB 600:RETURN
335 ZN$(VAL(ZN$))=ZN$
340 X1$=RIGHT$(Z$,LEN(Z$)-LEN(ZN$)):GOSUB800
342 IF X1$=""THEN ZN$(VAL(ZN$))="":RETURN
345 FORI=1TOAB:IF X1$=B$(I)THENR2$="OK":BC$(VAL(ZN$))=STR$(I):I=AB+1
350 NEXT
355 REM
360 IF R2$="OK"THEN TE$(VAL(ZN$))=Z$:R2$="":RETURN
365 PRINT"SYNTAX ERROR":ER=2:GOSUB 600:RETURN
370 RETURN
500 POKE VI+32,0:POKEVI+33,0:REM MENU
505 PRINT CHR$(144);CHR$(147)
510 PRINTCHR$(176);:FORI=1TO38:PRINTCHR$(96);:NEXT: PRINTCHR$(174):X=1:Y=23
515 GOSUB 5000: PRINTCHR$(173);:FORI=1TO38:PRINTCHR$(96);:NEXT:PRINTCHR$(189)
520 FOR I=1104TO1104+19*40STEP40:POKE I, 93:POKEI+54272,0:POKEI+39,93
525 POKEI+39+54272,0:NEXT
530 X=5:Y=4:GOSUB5000:PRINT"HOFER, METZGER * 1984
535 POKE1224,107:FORI=1225 TO 1225+38:POKEI,67:POKEI+54272,0:NEXT
537 POKE 1224+39,115:POKE1224+39+54272,0
538 X=4:Y= 9:GOSUB 5000:PRINT"■*** 2 PRESE DI CORR. PROGRAMMABILI ***■"
```

```
540 X=4:Y=12:GOSUB 5000:PRINT"INIZIALIZZAZIONE PROGRAMMA    1 "
545 X=4:Y=14:GOSUB 5000:PRINT"INIZIO PROGRAMMA      2 "
550 X=4:Y=16:GOSUB 5000:PRINT"SAVE, LOAD, PRINT     3 "
555 X=3:Y=20:GOSUB 5000:PRINT"■▲ PREMETE IL TASTO DESIDERATO! ■■"
557 POKE VI+32,6:POKEVI+33,6:PRINT"■"
560 GETX$: IFX$=""THEN560
565 IFX$<>"1"ANDX$<>"2"ANDX$<>"3"THEN560
570 ON VAL(X$)GOSUB 200,1000,2000
575 GOTO 500
600 REM AVVISO DI ERRORE
605 IFER=1THENPRINT:PRINT"SONO PERMESSE LINEE DA ■1■ A ■20■!
":PRINT:R=0:RETURN
610 IFER=2THENPRINT:PRINT"ORDINE■ ";X1$:PRINT"■NON RICONOSCIUTO!":PRINT:ER=
699 RETURN
800 REM PARAMETRI PER TEMPO E GOTO
805 IF LEFT$(X1$,4)="TEMPO"THEN817
810 IF LEFT$(X1$,4)="GOTO"THEN817
815 RETURN
817 IFLEFT$(X1$,4)="TEMPO"THENGOTO830
820 IFMID$(X1$,5,1)=""THEN RETURN
823 IFZN$(VAL(MID$(X1$,5,3)))=""THENPRINT:PRINT"INDIRIZZO
SALTO SCONOSCIUTO!"R2$ ="OK":RETURN
825 BC$(VAL(ZN$))="50 "+MID$(X1$,5,3):R2$="OK":RETURN
830 P$=MID$(X1$,5,1)
835 IFP$<>"1"ANDP$<>"2"ANDP$<>"3"THENRETURN
840 IFMID$(X1$,6,1)<>"="THENRETURN
845 BC$(VAL(ZN$))="51 "+MID$(X1$,5,1)+" "+MID$(X1$,7,3):R2$="OK":RETURN
1000 REM INIZIO PROGRAMMA
1005 PO=56576:POKEPO+3,7:POKEPO+1,0:PRINT"■"
1010 FORI=1TO20:GOSUB1700
1012 IFBC$(I)=" 16"THENI=21:GOTO1060
1020 IF ZN$(I)=""ORZN$(I)="*"THEN1050
1025 IF LEFT$(BC$(I),2)="50"ORLEFT$(BC$(I),2)="51"THEN1620
1030 ON VAL(BC$(I))GOTO1300,1310,1320,1330,1340,1350,1360,1390,1420,1450
1035 X=VAL(BC$(I))-10:ONXGOTO 1480,1510,1540,1570,1600
1040 GETX$:IFX$="■"THENI=21:GOTO1060
1050 REM
1060 NEXT:POKEPO+1,0:RETURN
1200 REM TEMPORIZZATORE
1205 REM
1210 IFQ=1THENFORK=P*1000TO0STEP-1:NEXT:Q=0:RETURN
1220 IFQ=2THENFORL=P*1000*50TO0STEP-1:NEXT:Q=0:RETURN
1230 IFQ=3THENFORM=P*1000*50*60TO0STEP-1:NEXT:Q=0:RETURN
1240 RETURN
1300 POKEPO+1,PEEK(PO+1)ANDNOT1:GOTO1040
1310 POKEPO+1,PEEK(PO+1)ANDNOT2:GOTO1040
1320 POKEPO+1,PEEK(PO+1)ANDNOT4:GOTO1040
1330 POKEPO+1,PEEK(PO+1)OR1:GOTO1040
1340 POKEPO+1,PEEK(PO+1)OR2:GOTO1040
1350 POKEPO+1,PEEK(PO+1)OR4:GOTO1040
1360 REM
1370 IFPEEK(PO+1)AND8THENPOKEPO+1,PEEK(PO+1)OR1:GOTO1040
1380 POKEPO+1,PEEK(PO+1)ANDNOT1:GOTO1040
1390 REM
1400 IFPEEK(PO+1)AND16THENPOKEPO+1,PEEK(PO+1)OR1:GOTO1040
1410 POKEPO+1,PEEK(PO+1)ANDNOT1:GOTO1040
1420 REM
1430 IFPEEK(PO+1)AND32THENPOKEPO+1,PEEK(PO+1)OR1:GOTO1040
1440 POKEPO+1,PEEK(PO+1)ANDNOT1:GOTO1040
```

```
1450 REM
1460 IFPEEK(PO+1)AND8THENPOKEPO+1,PEEK(PO+1)OR2:GOTO1040
1470 POKEPO+1,PEEK(PO+1)ANDNOT2:GOTO1040
1480 REM
1490 IFPEEK(PO+1)AND16THENPOKEPO+1,PEEK(PO+1)OR2:GOTO1040
1500 POKEPO+1,PEEK(PO+1)ANDNOT2:GOTO1040
1510 REM
1520 IFPEEK(PO+1)AND32THENPOKEPO+1,PEEK(PO+1)OR2:GOTO1040
1530 POKEPO+1,PEEK(PO+1)ANDNOT2:GOTO1040
1540 REM
1550 IFPEEK(PO+1)AND8THENPOKEPO+1,PEEK(PO+1)OR4:GOTO1040
1560 POKEPO+1,PEEK(PO+1)ANDNOT4:GOTO1040
1570 REM
1580 IFPEEK(PO+1)AND16THENPOKEPO+1,PEEK(PO+1)OR4:GOTO1040
1590 POKEPO+1,PEEK(PO+1)ANDNOT4:GOTO1040
1600 REM
1605 IFPEEK(PO+1)AND32THENPOKEPO+1,PEEK(PO+1)OR4:GOTO1040
1610 POKEPO+1,PEEK(PO+1)ANDNOT4:GOTO1040
1620 IF LEFT$(BC$(I),2)="50" THEN1670
1630 Q=VAL(MID$(BC$(I),4,1))
1640 REM
1650 P=VAL(MID$(BC$(I),6,3))
1660 GOSUB1200:GOTO1040
1670 I=VAL(MID$(BC$(I),4,2)):GOTO1012
1700 REM TEMPORIZZATORE
1705 PRINT"■                                    "     "
1710 PRINT"■PASSO DI PROGRAMMA:■ ";I;"■":RETURN
2000 REM SAVE, LOAD, PRINT
2005 PRINT"■■  SALVA IL PROGRAMMA  1"
2010 PRINT"  CARICA IL PROGRAMMA  2"
2015 PRINT"  STAMPA IL PROGRAMMA  3■"
2020 PRINT:PRINT"INSERITE IL NUMERO  M=MENU"
2030 GETX$:IFX$=""THEN2030
2035 IFX$="M"THENRETURN
2040 IFVAL(X$)<=0ORVAL(X$)>3THEN2030
2045 ON VAL(X$)GOSUB2100,2200,2300
2050 RETURN
2100 PRINT:PRINT"INSERITE NOME PROGRAMMA ";:X=8:GOSUB5500
2105 IF Z$=""THENRETURN
2110 X=0:FORI=1TO20:IFZN$(I)=""THENX=X+1
2115 NEXT:IF X=20 THENRETURN
2120 PRINT:PRINT"1=FLOPPY DISK":PRINT"2=CASSETTA"
2125 PRINT:PRINT"INSERITE IL NUMERO!   M=MENU":PRINT
2130 GETX$:IFX$="M" THENRETURN
2135 IF X$="1"THENFL=8
2140 IF X$="2"THENFL=0
2150 IF X$<>"1"ANDX$<>"2"THEN2130
2155 IF FL=8THENCLOSE1:OPEN1,8,2,"@:"+Z$+",S,W"
2157 IF FL=0THENCLOSE1:OPEN1,1,1,"@:"+Z$+",S,W"
2158 PRINT#1,Z$
2160 FORI=1TO20
2165 IF ZN$(I)=""THENZN$(I)="*":BC$(I)="*":TE$(I)="*"
2170 PRINT#1,ZN$(I)
2175 PRINT#1,BC$(I)
2180 PRINT#1,TE$(I)
2185 NEXT:CLOSE1
2190 FORI=1TO20:IFZN$(I)="*"THENZN$(I)=""
2195 :NEXT:RETURN
2200 PRINT:PRINT"INSERITE NOME PROGRAMMA ";:X=8:GOSUB5500
```

```
2205 IF Z$=""THENRETURN
2210 FORI=1TO20:ZN$(I)="":BC$(I)="":TE$(I)=""
2215 NEXT
2220 PRINT:PRINT"1=FLOPPY DISK":PRINT"2=CASSETTA"
2225 PRINT:PRINT"INSERITE IL NUMERO!    M=MENU":PRINT
2230 GETX$:IFX$="M" THENRETURN
2235 IF X$="1"THENFL=8
2240 IF X$="2"THENFL=0
2250 IF X$<>"1"ANDX$<>"2"THEN2230
2255 IF FL=8THENCLOSE1:OPEN1,8,2,+Z$+",S,R"
2257 IF FL=0THENCLOSE1:OPEN1,1,0,+Z$+",S,W"
2258 INPUT#1,PN$
2260 FORI=1TO20
2270 INPUT#1,ZN$(LIST2125
2275 INPUT#1,BC$(I)
2280 INPUT#1,TE$(I)
2282 IF ZN$(I)="*"THENZN$(I)=""
2283 IF BC$(I)="*"THENBC$(I)=""
2284 IF TE$(I)="*"THENTE$(I)=""
2285 NEXT:CLOSE1:RETURN
2300 PRINT"■STAMPANTE OK  S/N"
2310 GETX$:IFX$="N"THENRETURN
2315 IFX$="S"THEN2325
2320 GOTO2310
2322 X=0:FORI=1TO20:IFTE$(I)=""THENX=X+1
2323 IF X=20 THENRETURN
2325 CLOSE1:OPEN1,4
2330 PRINT#1,"LISTATO PROGRAMMA"
2335 PRINT#1,"PRESE DI CORRENTE PROGRAMMABILI":PRINT#1,
2337 IF PN$=""THEN INPUT"NOME PROGRAMMA:";PN$
2340 PRINT#1,"NOME PROGRAMMA:";PN$:PRINT#1,
2345 FORI=1TO20
2350 IF TE$(I)=""THEN2360
2355 PRINT#1,TE$(I)
2360 NEXT:PRINT#1,:PRINT#1,"READY":CLOSE1:RETURN
3000 DATA DOSE1=0,DOSE2=0,DOSE3=0,DOSE1=1,DOSE2=1,DOSE3=1
3005 DATA DOSE1=EINGANG1,DOSE1=EINGANG2,DOSE1=EINGANG3
3010 DATA DOSE2=EINGANG1,DOSE2=EINGANG2,DOSE2=EINGANG3
3015 DATA DOSE3=EINGANG1,DOSE3=EINGANG2,DOSE3=EINGANG3
3020 DATA END
4000 REM LIST
4005 PRINT"■"
4010 FORI=1TO20
4015 IF ZN$(I)=""THEN4030
4020 PRINTTE$(I)
4025 REM
4030 NEXT
4035 RETURN
5000 REM POSIZIONAMENTO CURSORE ********
5005 MS=40:MZ=25
5010 IFX>MSORY>MZTHENRETURN
5015 PRINT"■"
5020 IFX<=1THEN5040
5025 FORI=1TOX-1
5030 PRINT"■";
5035 NEXTI
5040 IFY<=1THENPRINT"□";:RETURN
5045 FORI=1TOY-1
5050 PRINT"■";
```

```
5055 NEXTI:PRINT"□";
5060 RETURN
5300 REM CURSORE ACCESO ****************
5305 PRINT" ";:FORT=1TO150:NEXT:RETURN
5350 REM CURSORE SPENTO ****************
5355 PRINT"_";:FORT=1TO150:NEXT:RETURN
5500 REM INPUT ***************************
5505 Z$=""
5510 GETX$:IFX$=""THENGOSUB5300:GOTO5520
5515 GOTO5525
5520 GETX$:IFX$=""THENGOSUB5350:GOTO5510
5525 REM SPAZIO LIBERO PER UNA CONDIZIONE
5530 IFX$="□"ORX$="■"ORX$="■"ORX$="□"ORX$="■"ORX$="■"THEN5510
5535 IFASC(X$)=13ANDLEN(Z$)=0THEN5510
5540 IFASC(X$)=20THENGOSUB5300:GOTO5560
5545 IFASC(X$)=13THENGOSUB5300:PRINT:RETURN
5550 IFLEN(Z$)>X-1THEN5510
5555 Y=PEEK(214):Y2=PEEK(211):PRINTX$;:Z$=Z$+X$:GOTO5510
5560 IFZ$=""THEN5510
5565 Z1$="":Z1$=LEFT$(Z$,LEN(Z$)-1):Z$=Z1$
5570 PRINT"■ ";:GOTO5510
5700 REM SUONO ***************************
5705 SI=54272:IFTL<=0THENRETURN
5710 POKESI+24,15:POKESI+5,0:POKESI+6,240
5715 POKESI,157:POKESI+1,69:POKESI+4,17:FORX=1TOTL:NEXT:POKESI+4,0
5720 RETURN
```

Il C64 come frequenzimetro

Il prezzo d'acquisto di un frequenzimetro è ancora abbastanza elevato, tuttavia in un laboratorio elettronico, anche da dilettanti, non può mancare uno strumento del genere. Siccome nel C64 è contenuto un contatore che può essere temporizzato attraverso la User-port, esso può essere utilizzato come un frequenzimetro vero e proprio. Contemporaneamente al conteggio della frequenza del segnale, viene fatto partire un timer che, alla fine della misura, ha memorizzato il tempo totale trascorso durante il processo di misura.
Con questo contatore si fa in modo che fondamentalmente la misura duri un secondo. Il risultato viene quindi ad essere sempre un risultato in Hertz. L'ingresso del segnale avviene attraverso l'ingresso CNT che è in grado di elaborare segnali con frequenza massima di circa 450 KHz.

Il programma

La misura della frequenza avviene attraverso il contatore interno A. Perciò nella linea 10 alla variabile PO viene assegnato l'indirizzo fondamentale 56576. Nella linea 20, il registro di controllo per il contatore A viene dapprima bloccato con l'istruzione POKE PO+14,8. Inoltre mediante l'assegnazione del valore 8, si comunica al timer che la sorgente del trigger del timer è costituita dal fronte di salita dell'impulso sul pin CNT. Viene poi azzerata l'ora: questo processo è importante nel caso che, durante la misura della frequenza, scatti l'ora interna. Poi il contatore viene caricato con i registri 4 e 5 uguali, nelle linee 30 e 40. Il valore 255 nei semi-byte inferiore e superiore pone il contatore a 65535 (ovvero 16 bit). Questa è anche la frequenza più alta che è possibile misurare. Un ampliamento della frequenza massima potrebbe essere effettuato utilizzando anche il contatore β e realizzando in tal modo un contatore a 32 bit. Siccome, per la misura di frequenze così alte, il procedimento di misura dovrebbe essere sicuramente scritto in linguaggio macchina, in questa sede non ci addentriamo in questo problema. Nella linea 50 vengono assegnati un contatore Z=Z+1 ed il tempo T1 uguale a TI determinando quindi il tempo d'inizio della misura. Viene poi attivato il registro di controllo nella linea 60 con POKE PO+14,1 per misurare l'impulso sull'ingresso CNT. Questo processo dura esattamente un secondo. La misura temporale deriva dall'istruzione FOR con la variabile temporanea K. Dopo di che viene fermato nuovamente il tempo con T2=TI nella linea 70, la quale blocca anche il contatore. Il risultato della misura della frequenza viene calcolato nella linea 80 con W1=65565 meno il contenuto del registro del contatore. Nella linea 90 viene determinata la frequenza dividendo il risultato W1 per il tempo trascorso. Questa procedura è importante nel caso che il processo temporale duri più a lungo di quanto determinato nell'istruzione FOR con la variabile K. La frequenza viene quindi stampata con la linea 100. Nella linea 110 si ritorna nuovamente alla 20 per la misura della frequenza. Per questo processo di misura viene quindi impiegato un secondo. Dopo viene mostrato il risultato, si misura per un altro secondo e così via.

```
2 REM C64 COME FREQUENZIMETRO
5 PRINT"■FREQUENZA (HZ)"
10 PO=56576
20 POKEPO+14,176:TI$="000000"
30 POKEPO+4,255
```

Il circuito

Poiché il procedimento di conteggio avviene sul pin CNT e questo ingresso riconosce soltanto il fronte di salita dell'impulso da contare, occorre preoccuparsi che il segnale da contare sia un segnale rettangolare ben formato. Di ciò si occupa il circuito integrato qui impiegato, SN74132. I diodi di ingresso D1 e D2 servono come diodi shunt in modo che sulla User-port, attraverso il segnale di ingresso, non possa arrivare un livello troppo alto. Il segnale da misurare viene formato attraverso il gate 1 del circuito integrato. Il segnale viene invertito attraverso l'uscita invertente e reinvertito poi mediante il gate 2. In questo modo, nell'ingresso CNT del contatore, entra un segnale rettangolare privo di rumore.

Elenco dei componenti del frequenzimetro:

D1 = 1N4148
D2 = 1N4148
IC1 = SN74132

```
POKEP0+5,255
Z=Z+1:T1=TI
POKEP0+14,177:FORK=1TO998:NEXT
T2=TI:POKEP0+14,8
W1=65535-(PEEK(P0+5)*256+PEEK(P0+4))
F=W1/((T2-T1)/60)
PRINT"[illegible]";F;"          "
GOTO20
```

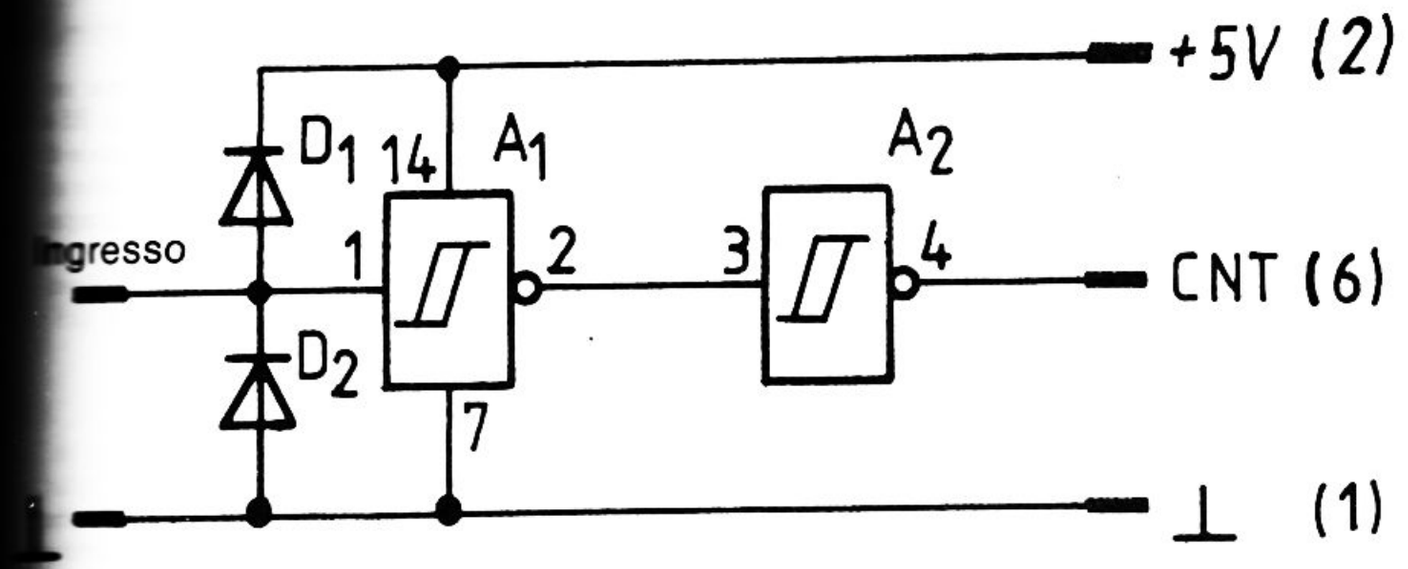


Fig. 1

Via sia luce

Il Commodore 64 non permette soltanto di accendere e spegnere la luce in una stanza, bensì permette anche di regolarne la luminosità mediante il controllo della tensione. Premetelo una volta e la stanza diventerà più luminosa, premetelo due volte e diventerà più buia. L'impiego di un circuito per l'attenuazione della luce mediante un calcolatore è certamente conveniente se questa attenuazione può essere ottenuta in seguito ad un comando esterno, come un rumore oppure la luminosità dell'ambiente ecc. L'attenuatore seguente mostra un esempio dell'attenuazione della luce per mezzo dei tasti o di un segnale d'ingresso sulla User-port.
La costruzione del circuito di attenuazione consiste di tre passi. Il primo passo consiste nella conversione digitale-analogica per la generazione da parte di un calcolatore di un segnale analogico. Il secondo consiste nell'accoppiamento dei segnali analogici mediante un fotoaccoppiatore. Il terzo modulo consiste di un circuito per il controllo della fase. Il convertitore digitale e il circuito di controllo in fase sono separati elettricamente dal fotoaccoppiatore. In questo modo la tensione di 220 Volt è sicuramente separata dal calcolatore.

Convertitore A/D

Il calcolatore fornisce sulla sua User-port un segnale digitale che può avere valore 0 o +5 Volt, ovvero stato 0 o 1. Un segnale analogico invece può avere qualunque valore intermedio, ad esempio 2,34 Volt.

Se volessimo regolare per esempio il numero di giri di un motore in corrente continua direttamente con il calcolatore, potremmo soltanto accenderlo o spegnerlo. Con un convertitore digitale-analogico esiste invece la possibilità di trasformare un segnale digitale in uno analogico e di mandare al motore questa tensione analogica. Aumentando o diminuendo questa tensione è quindi possibile regolare il numero di giri del motore stesso.

Convertitore D/A

Viene impiegato un convertitore digitale-analogico a 4 bit. Poiché ogni bit può assumere lo stato 0 oppure 1, vi sono perciò $2^4 = 16$ differenti combinazioni digitali a disposizione. Il convertitore D/A trasforma ora questi 16 segnali digitali in una corrispondente tensione analogica secondo la tabella seguente:

PB3	PB2	PB1	PB0	Tensione di uscita in Volt
0	0	0	0	0
0	0	0	1	0,5
0	0	1	0	1,0
0	0	1	1	1,5
0	1	0	0	2,0
0	1	0	1	2,5
0	1	1	0	3,0
0	1	1	1	3,5
1	0	0	0	4,0
1	0	0	1	4,5
1	0	1	0	5,0
1	0	1	1	5,5
1	1	0	0	6,0
1	1	0	1	6,5
1	1	1	0	7,0
1	1	1	1	7,5

Descrizione del circuito

L'amplificatore operazionale del circuito lavora come sommatore. PB0 è il bit meno significativo e quindi vale 2^0 mentre PB3 è il più significativo ovvero vale 2^3. Se ad esempio vi è un valore logico 1 su PB0 e sulle altre tre uscite un valore logico 0, la tensione di uscita è di 0,5 Volt come si può calcolare secondo la formula:

$$Vu=(VPB0*R5/R1)=5*1/10$$

Le resistenze vengono scelte in progressione: ovvero R1, R1/2, R1/4 e R1/8. Per la resistenza R5 è previsto un trimmer potenziometrico. Il segnale di uscita dell'amplificatore operazionale permette ad ogni valore di tensione l'accensione della lampada ad una tensione corrispondente. La tensione di alimentazione per l'operazionale è prelevata direttamente dal pin 2 della User-port.

Il fotoaccoppiatore

Utilizziamo un fotoaccoppiatore basato su di una resistenza fotosensibile. Questo fotoaccoppiatore può essere costruito in modo semplice ed economico. Occorre procurarsi un tubetto di plastica lungo dai 3 ai 4 centimetri con circa 11 mm di diametro interno. Da una parte viene fissata una lampadina ad incandescenza. Dall'altra parte, a distanza di circa due centimetri, viene fissata la resistenza fotosensibile. La costruzione precisa deve ora essere sigillata in modo che sia impenetrabile alla luce il che si può ottenere con un nastro isolante nero o, meglio ancora, chiudendola in una scatola nera.

Sezione per il controllo in fase

L'illuminazione della stanza con l'attenuatore deve avvenire senza salti dalla luce al buio. Mediante l'uso di un triac, questo circuito per il controllo della fase può essere realizzato a buon mercato e modo che tutte le uscite siano basse. Il programma vero e proprio comincia con la linea 60 e nelle

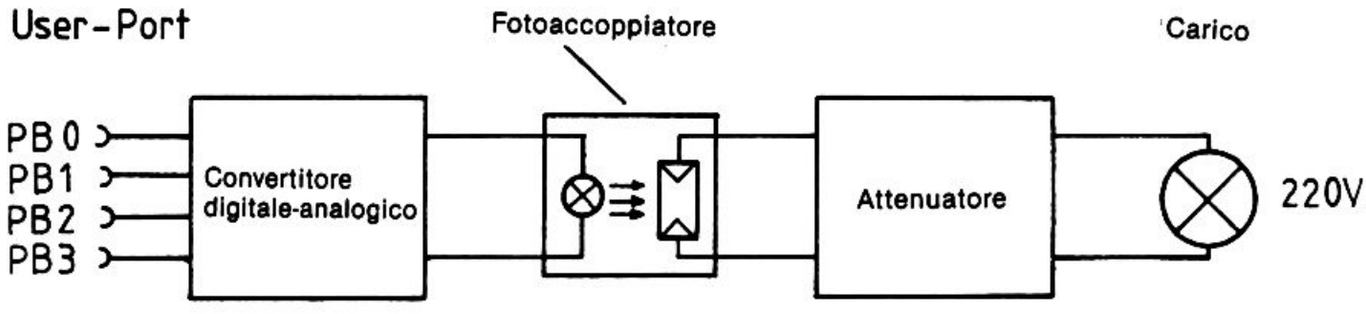


Fig. 1

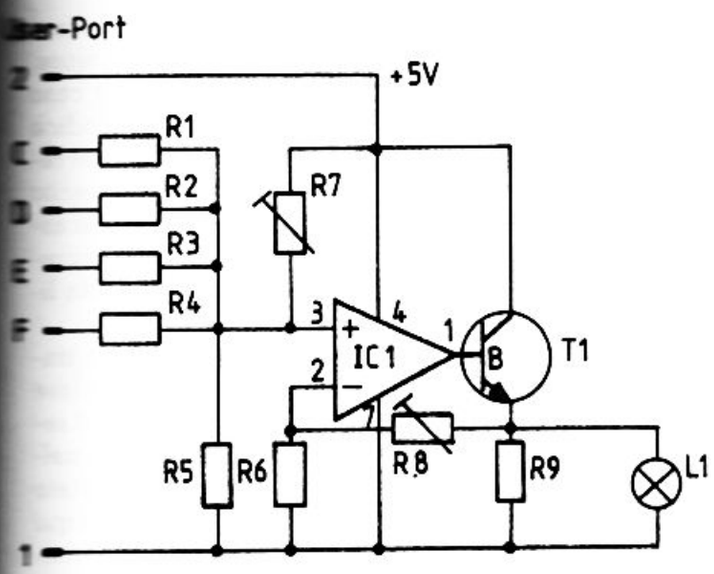


Fig. 2

di piccole dimensioni. Un triac è un interruttore elettronico che può essere aperto o chiuso come determinato dal suo circuito di comando.

Il circuito

Il carico (ovvero la lampadina) ed il triac sono collegati in serie con la tensione di alimentazione. Se il triac è in conduzione allora una corrente fluisce attraverso il carico. Il circuito di comando consiste di due circuiti RC e di un diac. All'inizio di ogni semionda, il condensatore C1 si carica attraverso le resistenze R1 ed R2. Non appena la tensione ai capi del condensatore C1 supera la tensione di scarica del diodo, questo entra in conduzione ed accende il triac. In funzione dello stato d'illuminazione della fotoresistenza, il tempo di carica del condensatore C1 è più corto o più lungo, e quindi il triac viene acceso prima o dopo.
Il secondo circuito RC è composto da R3 e C2. Questo circuito garantisce la costanza della condizione di accensione del triac. Senza di esso il condensatore C1 non verrebbe più caricato correttamente. Ciò significa che, se una pur piccola carica rimanesse sul condensatore in condizioni di riposo, alla prossima semionda il triac verrebbe acceso più rapidamente del previsto. Con il circuito descritto, la potenza applicata sul carico può variare dal 10 al 90 %.

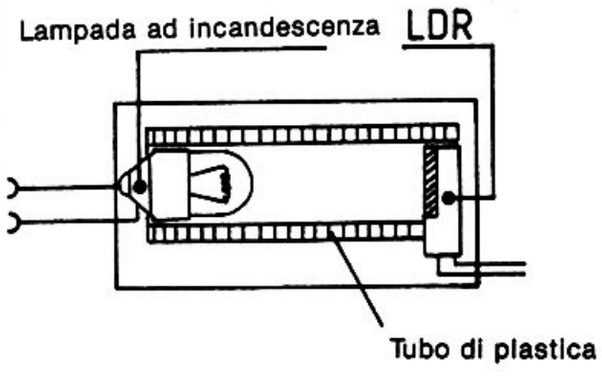


Fig. 3

Il montaggio

Per poter regolare la luce con il C64, occorre ora procedere al collegamento del convertitore digitale-analogico con il circuito per il controllo della fase. Questo collegamento viene effettuato per mezzo del fotoaccoppiatore. Si consiglia inoltre di chiudere il circuito per il controllo della fase in una scatola di plastica. Il connettore di terra (giallo-verde) deve essere collegato con la struttura metallica del carico. La maggior parte dei portalampada prevede a questo scopo un terminale cosiddetto terminale di massa.
L'inserimento di questa scheda in una scatola di plastica può essere facilitato usando un cavo di collegamento fissato alla scatola o alla piastra da un fermacavo.
Prima di lavorare con il circuito ricordatevi di togliere la tensione. Il triac impiegato, indicato nell'elenco dei componenti, può essere collegato con un carico massimo di circa 3 Ampere che corrispondono a circa 600 Watt. Se deve sopportare la potenza massima, il triac deve essere montato su un piccolo radiatore di alluminio.

Il programma

L'attenuazione della luce avviene attraverso i tasti funzione F1 e F7. Vi è inoltre la possibilità di regolare la luce per mezzo dell'interruttore sonoro o di altri segnali d'ingresso. Vengono usate le porte PB0-PB3 della User-port per il segnale digitale generato dal programma che viene inviato al circuito di attenuazione. Le porte della User-port vengono quindi usate come uscite del segnale. La luce viene regolata mediante un circuito OR. Avviene un rumore: la luce diventa più intensa. Avvengono due rumori: la luce diventa più bassa. Questa è la prima scelta OR. La seconda scelta OR si ha con i tasti di funzione F1 ed F7 già visti.

Il listato

Il programma si basa sulla generazione di un codice BCD sulle uscite PB0-PB3. L'ingresso PB7 serve per l'ingresso dell'interruttore sonoro. Le linee dalla 10 alla 30 regolano la User-port rispettivamente come uscita e come ingresso. Nella linea 30 il registro di uscita viene inizialmente azzerato in

linee seguenti si viene rinviati all'inizio del programma per mezzo di GOTO 60.
Dapprima nella riga 60 si assegna alla variabile X$ un carattere da tastiera. Segue quindi l'assegnazione del contenuto delle uscite PB0-PB3 alla variabile BCD. Nella linea 65 si prova se il tasto F3 è stato premuto. In questo caso il programma termina con l'istruzione END. Premendo il tasto F1, se lo stato del contatore BCD è minore di 15, il contenuto della User-port viene incrementato di 1 ed il risultato viene risalvato nello stesso registro. Ciò avviene nella linea 70. Nella linea 80 accade la stessa cosa ma viene interrogata la pressione del tasto F7 e se lo stato del contatore BCD è >0. In questo caso viene sottratto un 1 dal contenuto della porta PO+1 ed il risultato viene salvato nuovamente nella porta. In tal modo è possibile attraverso i tasti funzione F1 ed F7 regolare il codice BCD della User-port.

Richiesta

Nella linea 85 il contatore della variabile Z2 viene incrementato di 1. Questo contatore è importante per la richiesta del "doppio rumore". Poiché la pausa tra il primo ed il secondo rumore non può essere sempre regolare, occorre prevedere un determinato tempo di attesa. Si potrebbe anche usare un normale ciclo di ritardo con un'istruzione FOR, che terrebbe certamente occupato il calcolatore mettendo quindi fuori servizio i tasti funzione F1 ed F7. La richiesta dei tasti funzione F1 ed F7 viene fatta 8 volte. Se all'interno delle 8 richieste, l'ingresso sonoro ha due stati alti, questo segnale viene interpretato come un comando per l'attenuazione della luce ed elaborato di conseguenza dal programma. Se durante gli 8 cicli giunge un solo segnale sonoro, la luce diventa più intensa. In ogni caso il contatore Z2 viene azzerato, al più tardi quando il contatore ha raggiunto il valore 8. Supponiamo che vi sia stato un rumore: nella linea 95 il segnale 1 sul terminale PB7 della User-port viene assegnato alla variabile-registro. Ciò avviene con l'istruzione AND 128. Il registro ha ora il valore 1 ed il contatore Z è minore di 2 poiché finora soltanto un rumore è giunto. Nella linea 100 il contatore Z viene incrementato con Z=Z+1. L'istruzione FOR con la variabile K provoca ora una breve pausa di 0,5 secondi. Con ciò si aspetta la fine del segnale sonoro sulla User-port e viene azzerato il contatore Z2, dato che è iniziata una misura e si attende ora un secondo rumore. Se non si verifica alcun altro rumore, dopo al massimo altri 8 cicli, l'istruzione alla linea 120 viene eseguita. Il contatore Z2 è uguale a 8 ed il contatore Z1 ha valore 1; il codice BCD è inoltre minore di 15. In questo caso i contatori Z e Z2 vengono azzerati e la User-port viene nuovamente caricata con POKE (PO+1)+1. Si salta poi alla linea 60. Se il codice BCD è maggiore o uguale a 15, allora con un solo rumore non si verificherà alcun effetto. Se durante le 8 ripetizioni del ciclo si ha un altro rumore, allora viene eseguita la linea 110. In questo caso il contatore Z deve essere uguale a 2 ed il codice BCD deve essere diverso da 0. Il contenuto del registro PO+1 viene decrementato di 1 e reinserito nel registro stesso. Il contatore Z2 viene quindi azzerato nuovamente e si salta alla linea 60. L'incremento del contatore Z avviene ogni volta nella linea 100.

Il ciclo FOR della linea 100 può essere aggiustato con la variabile K rispettivamente al valore 500. Inoltre il tempo di attesa tra il primo ed il secondo rumore nella linea 120, (con IF Z=8) e nella linea 130 (con IF Z2>8), può essere aumentato o diminuito di un determinato fattore. I numeri qui impiegati sono il risultato di una prova sperimentale.

Elenco dei componenti del convertitore digitale-analogico:

R1	=	10 KOhm
R2	=	4,7 KOhm
R3	=	3,3 KOhm
R4	=	1,5 KOhm
R5	=	220 Ohm
R6	=	3,3 KOhm
R7	=	10 KOhm
R8	=	10 KOhm
R9	=	1 KOhm
T1	=	BC141
IC1	=	LM324
L1	=	6V/20mA

Attenuatore:

R1	=	10 KOhm
R2	=	LDR 05
R3	=	3,3 KOhm
R4	=	10 KOhm
C1	=	0,22 microF 250 Volt
C2	=	0,22 microF 250 Volt
D1	=	Diac D30 o ER900
Tr1	=	Triac TIC206D.

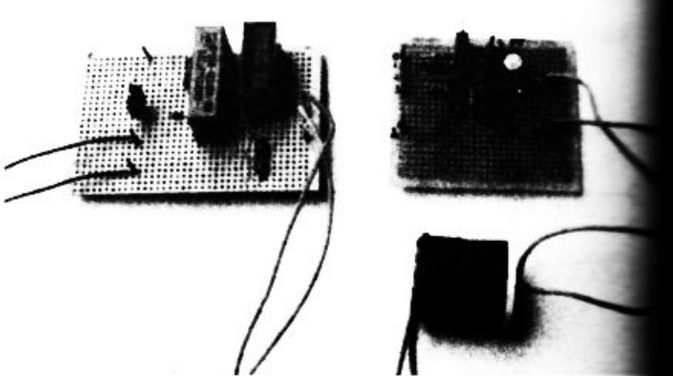
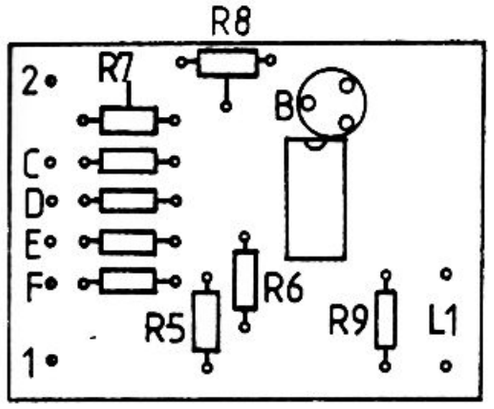
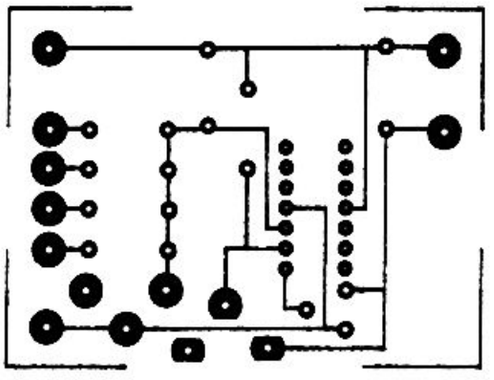


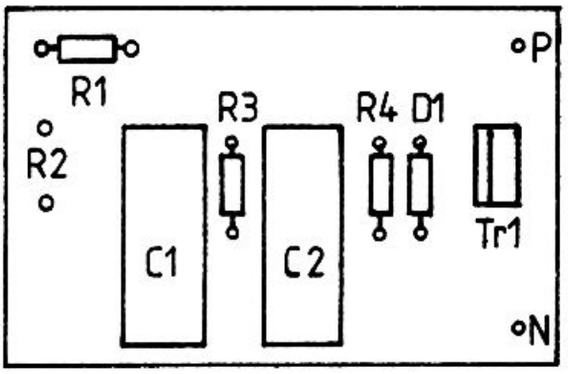
Fig. 4

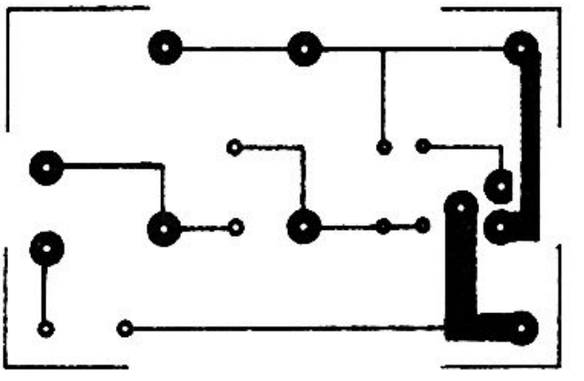




Convertitore digitale-analogico

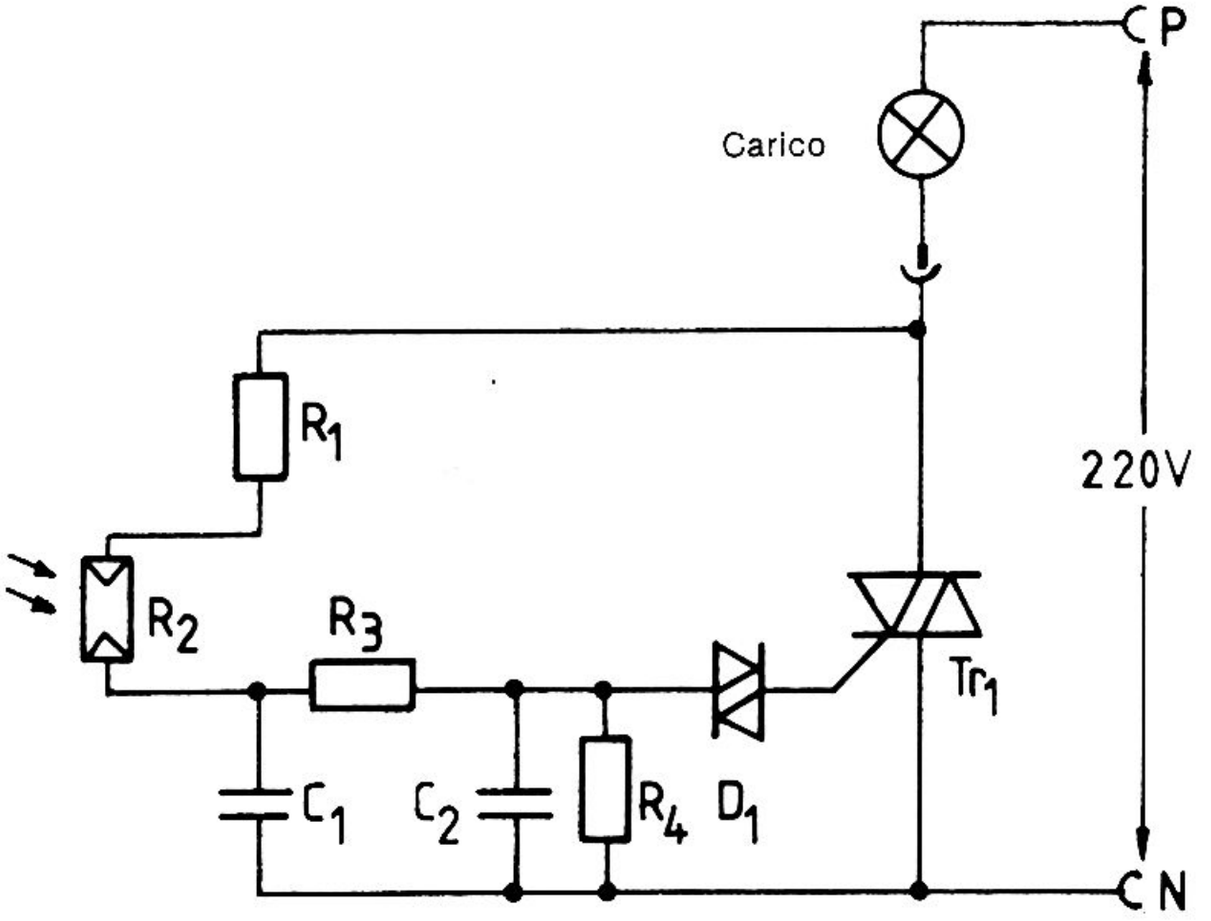
Fig. 5





Attenuatore

Fig. 6



```
5 REM VI SIA LUCE
10 PO=56576
20 POKEPO+3,63
30 POKEPO+1,0
40 POKE650,128 :REM RIPETITORE-FUNZIONE
50 REM ATTENUA CON F1 E F7
55 PRINTCHR$(147):PRINT"F3 = FINE!":PRINT
60 GETX$:BCD=PEEK(PO+1)AND15
65 IFX$=CHR$(134)THEN END
70 IFX$=CHR$(133)ANDBCD<15THENPOKEPO+1,(PEEK(PO+1)AND15)+1
80 IFX$=CHR$(136)ANDBCD<>0THENPOKEPO+1,(PEEK(PO+1)AND15)-1
85 Z2=Z2+1
90 REM ATTENUA SU PB7
95 REGISTER=PEEK(PO+1)AND128
100 IFREGISTER>1ANDZ<2THENZ=Z+1:FORK=1TO500:NEXT:Z2=0
AND15)-1:Z2=0:GOTO60
110 IFREGISTER>1ANDZ=2ANDBCD<>0THENZ=0:POKEPO+1,(PEEK(PO+1)
120 IFZ2=8ANDZ=1ANDBCD<15THENZ=0:Z2=0:POKEPO+1,(PEEK(PO+1)AND15)+1:GOTO60
130 IFZ2>8THENZ2=0:Z=0
140 GOTO60
```

Vita da cani

Senza un sensore ovvero senza una sorgente di segnali, la User-port non è di nessun aiuto per il comando di dispositivi esterni. È necessaria una sorgente di segnali che può essere ad esempio un sensore di temperatura o di umidità, un interruttore e così via. Proseguiamo spiegando il funzionamento di una barriera luminosa a raggi infrarossi ed il relativo progetto. L'utilizzazione di questa barriera luminosa può essere usata ad esempio per le prese di corrente programmabili. In questo modo il vostro quadrupede non avrà più alcuna possibilità di soffermarsi o entrare inosservato in una stanza vietata. Fondamentalmente la barriera luminosa rappresenta in questo contesto una generica sorgente di segnali.

Il circuito

I raggi infrarossi sono invisibili all'occhio umano. Il diodo trasmettitore D1 emette questo tipo di raggi infrarossi quando è attraversato da una corrente ovvero quando è collegato con l'anodo al positivo e col catodo al negativo; la corrente che provoca l'emissione di luce viene determinata dalla resistenza R1. Come ricevitore vi è un fototransistore Tr1; quando i raggi emessi dal diodo trasmittente giungono sull'area fotosensibile del fototransistore, aumenta la corrente di base nel diodo IB. La corrente di collettore del transistore è quindi funzione del flusso luminoso, della corrente di base e del fattore di amplificazione della corrente. Poiché questi non possono essere aumentati a piacere, occorre tendere alla più alta possibile sensibilità luminosa del diodo di base. Quest'alta sensibilità viene ottenuta facendo lavorare il fototransistore in configurazione a base aperta. Veniamo ora al circuito. Se la barriera luminosa non è interrotta, i raggi infrarossi giungono al transistore il quale avrà una corrispondente corrente di collettore. Questa corrente passa atraverso la resistenza R2 e, secondo la legge di Ohm, si ha una caduta di tensione di circa 4,7 Volt (V2=4,7 KOhm x 1mA).
Quando la barriera luminosa viene interrotta, nessun raggio luminoso può più giungere al transistore e la corrente di collettore si avvicina alla corrente di buio (circa $20*10^{-9}$A). Sulla resistenza R2 cade quindi una tensione prossima a 0 Volt.
Riassumendo: se la barriera non è interrotta, la tensione ai capi di R2 è di circa 4,7 Volt, se la barriera è interrotta la tensione su R2 è quasi nulla.
Poiché il segnale sul punto E2 rappresenta un determinato valore in Volt dato dalla caduta di tensione su R2, occorre fare in modo che il segnale venga inviato al calcolatore con una determinata tensione. Il computer ha bisogno infatti di stati per-

ettamente definiti come gli stati 0 oppure 5 Volt. Questi vengono ottenuti mandando la tensione ai capi di R2 da modificare sull'ingresso di un trigger di Schmitt. Un trigger di Schmitt è un interruttore a due valori, il cui potenziale di uscita viene determinato da un ben definito valore di tensione in logica 1. La tensione d'ingresso, detta anche tensione di soglia, è per il circuito integrato IC74132 di circa 1,7 Volt.
Il circuito integrato IC 74132 contiene quattro trigger di Schmitt invertenti. L'inversione della tensione di uscita è necessaria, poiché in questo modo quando la barriera luminosa è interrotta viene prodotto un impulso positivo.
Il diodo trasmettitore lavora con una corrente di circa 100 mA che corrisponde alla corrente massima che la User-port è in grado di fornire. Perciò è necessario utilizzare un generatore di tensione esterno. Per la realizzazione di un siffatto alimentatore, consultate il capitolo "Alimentazione". Con il generatore di tensione ivi impiegato, si possono accendere in tutto 8 barriere luminose. La sensibilità ai disturbi di questa barriera è inferiore a quella di una normale barriera luminosa con luce visibile. Per una barriera luminosa assolutamente insensibile ai disturbi occorrerebbe in realtà usare luce modulata. L'utente del calcolatore ha tuttavia la possibilita, con l'aiuto dei suoi programmi, di realizzare un filtro. Gli impulsi che la barriera luminosa ha inviato alla User-port possono essere conservati mediante un ciclo temporale e possono raggiungere in questo modo la porta in un ambito ben definito, prima che la porta intraprenda qualsiasi azione. In questo modo si ha la cosiddetta filtratura dei picchi senza che ci sia necessità di un più complicato circuito elettronico. Esaminate ora il listato del seguente programma.

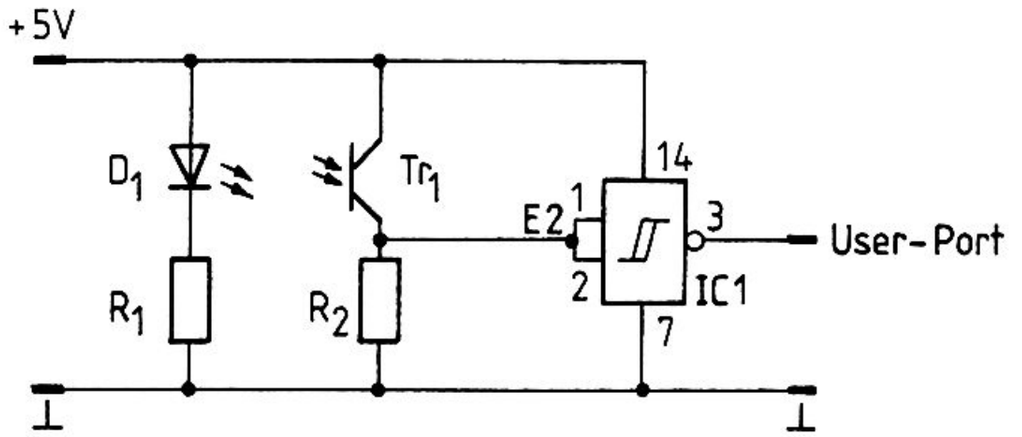


Fig. 1

```
5 REM VITA DA CANI
10 REM FILTRO
20 PO=56576
30 I=PEEK(PO+1)AND1:REM PORTA P60
40 IF I=1 THEN FORK=1TO20:NEXT
50 IF I=0 THEN 30
60 REM INGRESSO STABILIZZATO
70 REM CONTINUA IL PROGRAMMA
```

Il C64 come robot da tavolo

Il comando di dispositivi esterni, di macchinari e anche di intere fabbriche, può essere fatto per mezzo del C64 senza altre spese. A questo scopo naturalmente è necessaria un'interfaccia che colleghi il C64 con l'ambiente periferico.
Un'interfaccia di questo tipo è offerta dalla Fisher-Technik sotto il nome di Fisher-Technik-Computing. Questa è una scatola di montaggio che permette la costruzione di 6 robot professionali, terminali automatici e grafici per autocostruttori. Per mezzo dell'home computer, è possibile comandare, ovvero regolare, questi dispositivi. Nell'esempio seguente, descriveremo la costruzione di una macchina per selezionare oggetti comandata dal C64. Consideriamo dapprima il funzionamento generale di questa macchina. Osservate la figura. Su una traversa si trova il motore M1 che trasporta la slitta lungo un binario a cremagliera ed è così in grado di trasportare un blocco da costruzione da destra a sinistra e viceversa. Agli opposti terminali della slitta si trovano gli interruttori di fine corsa E1 ed E2. L'interruttore di fine corsa E3 si trova immediatamente a sinistra di E1 e serve per il posizionamento a riposo della slitta. Il posizionamento a riposo è predisposto in modo tale che un blocco da costruzioni possa essere ricevuto dalla piccola apertura sulla slitta. In alto a sinistra vi è l'interruttore E6 che serve come tasto di partenza dato che non registra il passaggio di blocchi da costruzione.

Selezionare con il C64

Lo scopo della macchina selezionatrice è il seguente. Dopo la pressione del tasto di partenza la slitta si sposta verso la posizione nella quale viene provocato il riconoscimento di un blocco piccolo oppure di uno grande mediante gli interruttori finecorsa E4 e E5. Se sulla slitta si trova un blocco grande, entrambi gli interruttori sono premuti, se vi si trova un blocco piccolo allora viene dapprima premuto l'interruttore E5 e successivamente l'E4. Se invece nessun blocco si trova sulla slitta, questa si sposta fino all'interruttore E2 e torna poi alla posizione di partenza all'interruttore E3.
Risultato: il funzionamento della macchina per la selezione consiste unicamente nel selezionare oggetti mediante le condizioni degli interruttori E4 ed E5. Tutti gli altri interruttori sono usati per definire i limiti del movimento della slitta.
Il comando può ora essere generato sulla User-port e per questo abbiamo bisogno di un programma. Prima però di procedere alla sua realizzazione, occorre che il programmatore abbia chiaro in quale modo debba essere gestita la logica per il comando della macchina selezionatrice. Si può iniziare prima di tutto da ciò, dato che, per l'utente non esperto, un tale compito può risultare difficile.
Noi vogliamo superare completamente questa situazione e perciò la problematica di comando verrà analizzata a fondo. Terminata quest'analisi, potremo iniziare a scrivere il programma, cominciando dalle singole funzioni. Non soltanto la preparazione del programma costituisce un problema, ma anche l'elaborazione del segnale del Commodore per la macchina selezionatrice. A questo scopo abbiamo realizzato una interfaccia costituita da 8 ingressi ed 8 uscite che può anche essere ampliata. Tuttavia per un semplice compito di comando, questa interfaccia è di gran lunga sufficiente. Elenchiamo ora tutti i segnali importanti per il pilotaggio della macchina selezionatrice:

Vi sono i seguenti interruttori di fine corsa ovvero i seguenti segnali di ingresso:

E1 = Slitta a destra
E2 = Slitta a sinistra
E3 = Slitta in posizione di riposo
E4 = Segnale di riconoscimento grande/piccolo
E5 = Segnale di riconoscimento grande/piccolo
E6 = Tasto di partenza.

I segnali di uscita per il comando della selezionatrice sono i seguenti:

M1 = Motore per lo spostamento della slitta
L1 = Lampada 1 per la selezione piccolo
L2 = Lampada 2 per la selezione grande.

Riassumendo abbiamo sei ingressi e tre uscite. Cominciamo ora con l'analisi delle varie funzioni.

Spostamento della slitta

Per l'analisi del funzionamento occorre suddividere il funzionamento generale in singole funzioni. Abbiamo perciò bisogno del cosiddetto albero delle funzioni che non è altro che una tabella, nella quale vengono indicate tutte le funzioni derivate da una funzione principale.
Una funzione ben chiara è il movimento della slitta. Cominciamo con il chiarire lo scopo delle sin-

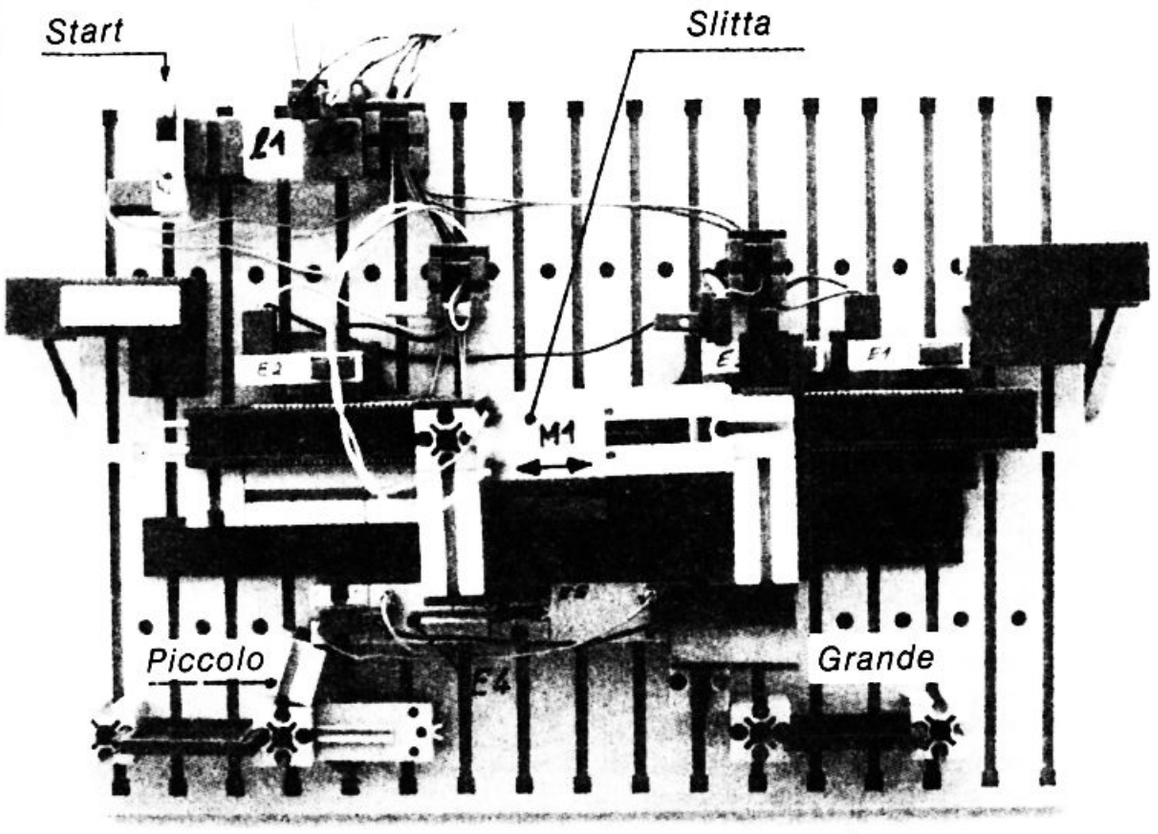


Fig. 1

gole funzioni in un ordine qualunque. Come già visto, abbiamo le funzioni Spostamento della slitta verso destra e Spostamento della slitta verso sinistra. La funzione Slitta stop indica che la slitta non si muove. Questa è ovviamente una cosiddetta funzione negativa, al contrario delle funzioni Slitta verso destra e Slitta verso sinistra. Se la slitta non si muove, logicamente essa deve restare ferma ed allora le funzioni Slitta a destra e Slitta a sinistra sono interrotte. La funzione Slitta stop in questo caso rappresenta quindi una funzione supplementare che, considerata logicamente, ha lo stesso significato della disattivazione delle funzioni Slitta verso destra e Slitta verso sinistra. Un'altra funzione è costituita dall'avvio del processo con il tasto Start. Chiamiamo questa funzione "Inizio selezione". Dopo che la slitta ha iniziato a muoversi verso sinistra, è possibile, dallo stato degli interruttori E4 ed E5, determinare se sulla slitta stessa si trova un blocco piccolo, uno grande oppure nessun blocco. Chiamiamo questa funzione "Selezione". Dopo che il dispositivo ha riconosciuto se è presente qualche blocco, il motore deve continuare a muoversi nella direzione corrispondente, altrimenti viene disinserito. L'accensione e lo spegnimento del motore sono stati già descritti nell'ambito delle funzioni Slitta verso sinistra e Slitta verso destra.

Dopo il riconoscimento avvenuto nel processo di selezione, occorre accendere le lampade L1 o L2. Il movimento della slitta verso destra ovvero verso sinistra potrebbe essere indicato anche da queste lampade, ma ciò non è poi così interessante dato che il movimento della slitta può essere osservato direttamente. Le lampade L1 ed L2 devono quindi accendersi soltanto quando si è effettuata una selezione grande o piccolo. La slitta quindi si muoverà verso sinistra fino all'interruttore E2 oppure verso destra fino all'interruttore E1, e lì si fermerà. Con questa funzione di stop si può anche spegnere la lampada che ha indicato il risultato della selezione. La caduta del blocco è automaticamente determinata dalla costruzione meccanica del dispositivo di selezione.

Terminata la selezione, la slitta deve tornare alla posizione di partenza. Il prossimo compito consiste ora nello studiare le singole funzioni e costruire quindi il processo logico. Poiché nell'assegnazione degli incarichi abbiamo a che fare con segnali degli interruttori di fine corsa, è assolutamente indispensabile determinare una certa sequenza di comandi con una data tecnica. La tecnica più facile, e tuttavia non molto chiara, sarebbe la personale euforia che nasce proprio in questo momento. Esempio: quando la slitta si sposta verso sinistra e gli interruttori E4 ed E5 sono premuti allora la slitta deve spostarsi nuovamente verso destra. In caso contrario la slitta deve comunque spostarsi

verso sinistra. Nascono così più funzioni per lo spostamento verso destra e verso sinistra.
Tutte queste situazioni devono essere riassunte in un cosiddetto prospetto di scelte OR e le rispettive uscite devono essere comunicate al motore. Per assolvere questi incarichi, abbiamo deciso di utilizzare la cosiddetta tecnica del circuito di corrente, che è la più facile e comprensibile. In questa tecnica, gli interruttori di fine corsa vengono indicati come interruttori ed il carico viene rappresentato come avvolgimenti. L'avvolgimento rappresenta in questo caso la bobina di un relè oppure direttamente l'avvolgimento di una lampadina ad incandescenza. Consideriamo ad esempio un circuito di corrente che rappresenta una semplice funzione logica.

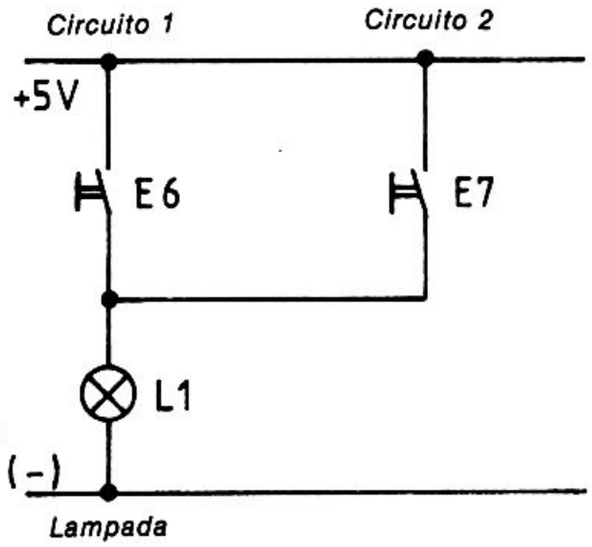


Fig. 2

Il circuito di corrente

Il circuito di corrente indicato nella figura seguente, rappresenta la funzione "Lampada". Nella parte superiore del circuito di corrente si trova la sorgente ovvero gli alimentatori dei singoli circuiti. Il collegamento di questi alimentatori, in questo caso +5 Volt, al negativo ovvero anche allo zero, rappresenta un circuito di corrente che genera un cortocircuito. Nel nostro esempio, abbiamo un circuito di corrente costituito da un tasto E6 e da una lampada L1. Come indicato nel disegno, la lampada L1 è spenta. La motivazione di ciò sta nel fatto che il circuito di corrente è interrotto dall'interruttore E6. Quindi non può circolare alcuna corrente e la lampada è perciò spenta. Premendo il tasto E6, il circuito viene chiuso e la lampada si accende. Questa è una semplice spiegazione di un semplice circuito di corrente. Nel disegno seguente la lampada L1 viene accesa anche quando un altro tasto E7 viene premuto. Perciò occorrono 2 circuiti di corrente che si riuniscono poi, usando entrambi lo stesso carico. La lampada ora si può accendere quando viene premuto il tasto E6 oppure il tasto E7. Lo stesso avviene ovviamente quando entrambi sono premuti. Basandoci su questa teoria, studieremo ora le singole funzioni per il comando della slitta. Da ciò costruiremo poi il programma corrispondente.

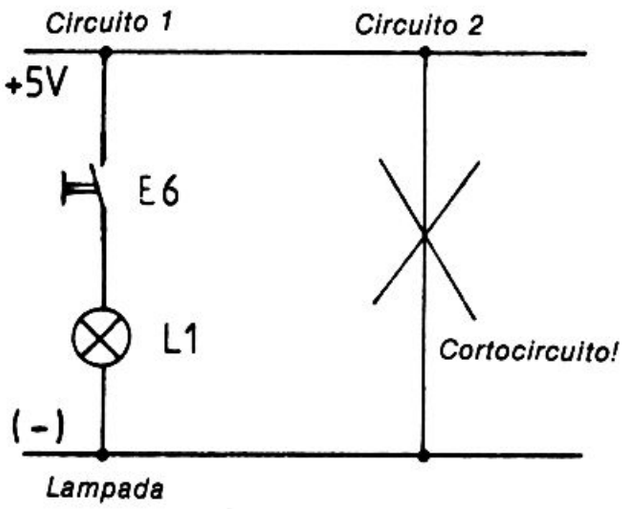


Fig. 3

Elenco delle funzioni da studiare:

Grande/piccolo
Posizione di partenza
Inizio selezione
Slitta a sinistra Q1
Slitta a destra Q2
Lampada grande L1
Lampada piccolo L2

Blocco grande e blocco piccolo

Gli interrutori E4 ed E5 vengono utilizzati per la funzione grande/piccolo. Un blocco grande preme contemporaneamente entrambi gli interruttori mentre uno piccolo preme gli interruttori in successione uno dopo l'altro. Il risultato di questa funzione è il comando del motore M1 in direzione destra o sinistra. Poiché dopo la pressione del tasto di inizio la slitta deve muoversi in ogni caso, essa deve dirigersi dapprima verso sinistra per portare il blocco nella posizione corrispondente agli interruttori E4 ed E5 e, poiché con un blocco piccolo i due interruttori non verranno mai premuti contemporaneamente, si può affermare che la slitta deve andare verso sinistra finché entrambi gli inter-

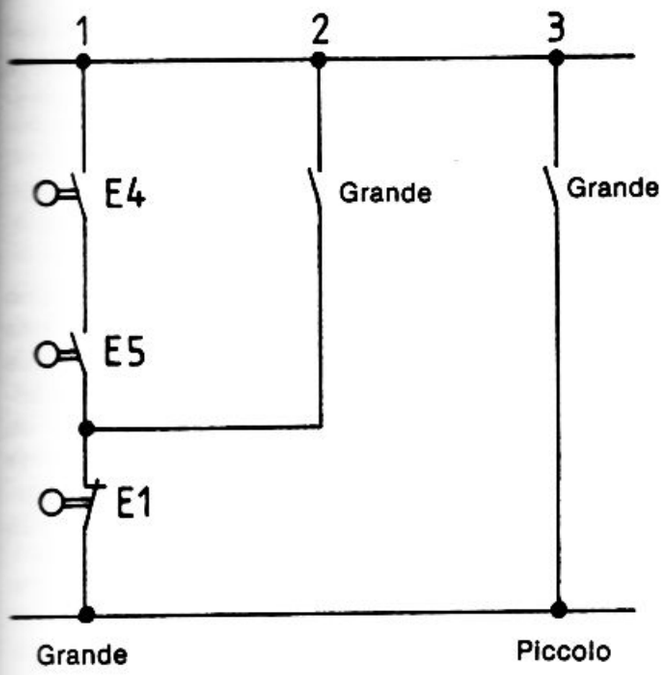
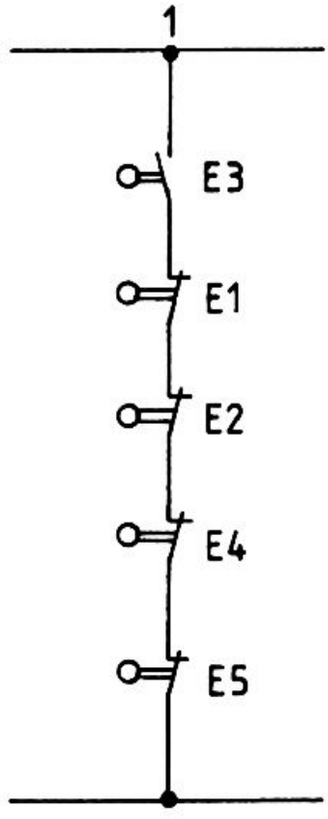


Fig. 4

ruttori non vengono premuti contemporaneamente. La funzione "Piccolo" è esattamente l'opposto della funzione "Grande". Diciamo quindi: se un blocco grande si trova sulla slitta, allora entrambi gli interruttori sono premuti. In caso contrario si tratta di un blocco piccolo.
Se nessun blocco si trova sulla slitta, questo viene interpretato come un blocco piccolo. In entrambi i casi la slitta si muove verso sinistra fino all'interruttore E2.
Dato che molte altre funzioni spostano la slitta verso destra o verso sinistra, la funzione grande/piccolo deve essere dapprima considerata come una funzione semplice. Se entrambi gli interruttori sono premuti allora si è in presenza del messaggio "Grande". In questo caso la slitta deve cambiare il suo senso di rotazione e muoversi verso destra. Poiché ora il blocco lascia l'interruttore anche il messaggio "Grande" non esiste più. Perciò la slitta dovrebbe cominciare a muoversi verso sinistra. Allora il blocco premerebbe nuovamente i due interruttori e la slitta si muoverebbe nuovamente verso destra. Il processo continuerebbe a ripetersi e ciò chiaramente non ha alcun senso.
Il segnale "Grande" va quindi conservato e cancellato in seguito con un segnale. La conservazione del segnale avviene attraverso entrambi gli interruttori E4 ed E5. Quando questi sono chiusi, la corrente arriva all'interruttore E1 e poiché questo non è stato toccato, la corrente passa e soddisfa quindi la funzione "Grande". Abbiamo quindi ora il messaggio "Grande" e possiamo inserire, mediante questo messaggio, un altro interruttore in parallelo ad E4 ed E5. Questo interruttore sta quindi per "Grande" e conserva questo messaggio. Quando poi la slitta si trova nella posizione finale E1 ed il blocco è caduto,la funzione "Grande" viene cancellata.
Un nuovo inserimento è nuovamente possibile quando gli interruttori E4 ed E5 sono premuti. Analogamente avremo la funzione "Piccolo", quando la funzione "Grande" non è attiva. Nel terzo circuito di corrente la funzione "Piccolo" è rappresentata con un cosiddetto dispositivo di apertura che deriva dalla funzione "Grande". Se questo è chiuso, allora abbiamo la funzione "Piccolo".
Abbiamo rinunciato alla rappresentazione di una bobina o più tecnicamente di un carico. Nei circuiti da realizzare si instaura quindi, al verificarsi delle condizioni, un cortocircuito. I nostri carichi sono in realtà le celle di memoria del nostro calcolatore e perciò non siamo stati molto rigorosi.
La posizione di partenza del dispositivo di selezione viene raggiunta quando la slitta si trova sull'interruttore E3 e gli interruttori E1, E2, E4 ed E5 non sono premuti. Se un interruttore è bloccato, ovvero per qualsiasi motivo è premuto, allora la posizione di partenza del dispositivo non è garantita e non è possibile avviare la slitta. Naturalmente queste considerazioni possono essere anche tralasciate, ma noi vogliamo avere un risultato corretto. Un contatto errato di un interruttore difettoso deve essere certamente eliminato.



Posizione iniziale

Fig. 5

Inizio e selezione

L'avvio della macchina avviene attraverso il tasto E6. È possibile avviare la macchina soltanto quando la posizione di partenza è garantita. Questo viene rappresentato nel circuito di corrente dal dispositivo di chiusura. Cominciamo quindi, per tutte le condizioni, con il riunire insieme in una locazione di memoria le funzioni totali costituite da più ingressi ed uscite.
Questa locazione di memoria comunica poi se la corrispondente funzione è soddisfatta o no. Se siamo nella posizione di partenza, allora l'interruttore della posizione iniziale è chiuso. Se la condizione non è soddisfatta, allora l'interruttore è aperto. Utilizziamo in un circuito di corrente un dispositivo di apertura come già successo nell'esempio Grande/piccolo con E1, soltanto che ora è invertito. Esempio: se volessimo usare in un circuito di corrente per la posizione iniziale un dispositivo di apertura, allora una corrente potrebbe passare nel circuito soltanto quando questo dispositivo è chiuso e quindi la posizione iniziale non è stata raggiunta. Dopo che il tasto di inizio E6 è stato premuto e la posizione iniziale della slitta è stata accertata, la slitta si muove verso sinistra e rilascia quindi l'interruttore E3. In tal modo la premessa fondamentale per il segnale della posizione iniziale non è più garantita. Il dispositivo di chiusura del nostro circuito si aprirebbe interrompendo la funzione "Inizio selezione". Anche il segnale di Inizio selezione deve quindi essere salvato. Se la slitta si trova nella posizione E2, il processo di selezione è terminato e la funzione "Inizio selezione" può essere disattivata. Osservate perciò il circuito corrispondente.

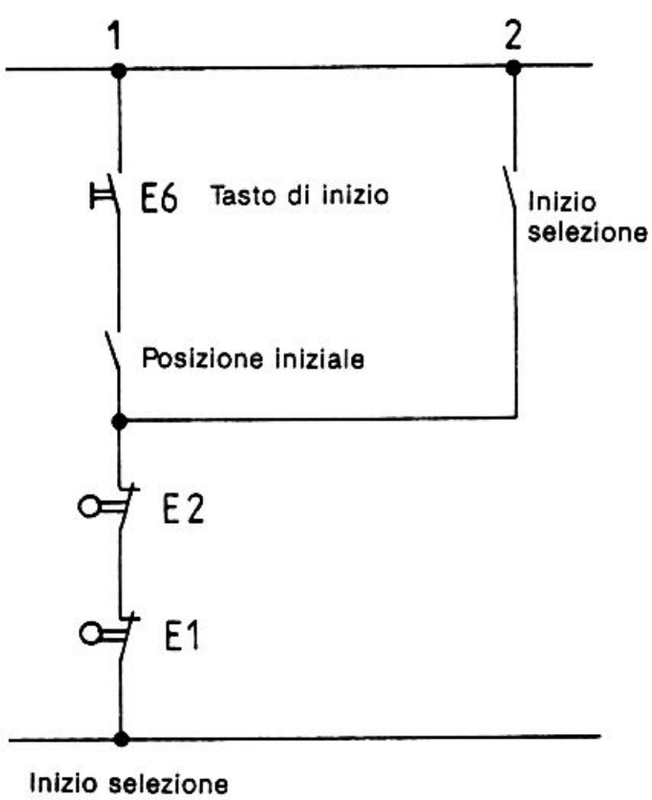


Fig. 6

Inizio del movimento

Lasciando ora la posizione iniziale, consideriamo ora la funzione "Slitta verso sinistra" e premiamo il tasto di inizio E6. Riceviamo quindi il segnale di inizio della selezione e la slitta comincia a muoversi verso sinistra. Trasportiamo dapprima un blocco piccolo e riceviamo, attraverso la funzione "Selezione", il messaggio "piccolo" poiché il messaggio "grande" non è stato attivato. La slitta può quindi muoversi ancora verso sinistra finché non viene fermata dall'interruttore E2. Poiché questo interruttore cancella anche il messaggio di "Inizio selezione" precedentemente memorizzato, anche l'interruttore Inizio selezione viene aperto. Mediante questo circuito inoltre, anche il motore non viene più trasportato verso sinistra. Finora tutto in ordine. Per sicurezza, in questo circuito, il dispositivo di apertura per "Slitta a destra" è ancora in-

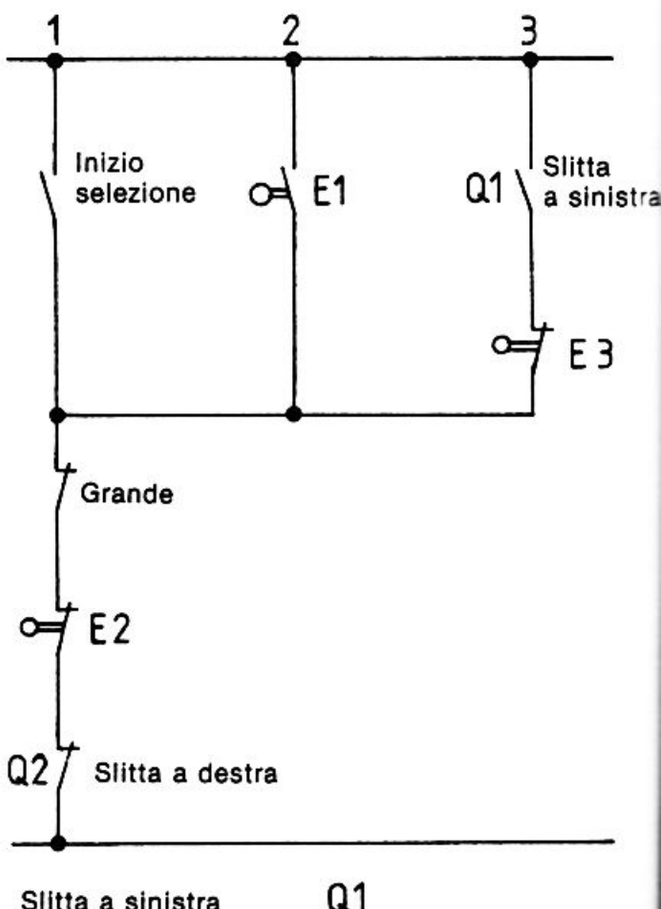


Fig. 7

serito. Nella funzione "Slitta a destra" è invece inserito un dispositivo di apertura "Slitta a sinistra". Ciò viene chiamato sbarramenbto incrociato. Siccome la slitta con una selezione "Grande" viaggia verso destra ed arriva all'interruttore E1, deve poi spostarsi verso sinistra per tornare alla posizione di partenza. Abbiamo quindi un'altra funzione per lo spostamento a sinistra e dobbiamo generare perciò un secondo circuito di corrente, parallelo al primo e che comandi anch'esso il motore verso sinistra. La slitta si trova quindi sull'interruttore E1 ed allora il motore viene comandato verso sinistra attraverso il secondo circuito ed il contatto Slitta a sinistra viene memorizzato sul terzo circuito. Questa condizione viene mantenuta fintantoché la slitta non raggiunge l'interruttore E3, ovvero la posizione iniziale. Il dispositivo di apertura del circuito 3 si apre e quindi viene cancellata anche la memorizzazione, dato che E1 non può più condurre.

Viceversa

La slitta si trova nella parte sinistra sull'interruttore E2. Il segnale "Slitta verso sinistra" è caduto e perciò l'interruttore E2 del circuito di corrente è interrotto. Una nuova attivazione non è possibile poiché anche la funzione di inizio è stata cancellata da E2 e l'attivazione sarà possibile soltanto

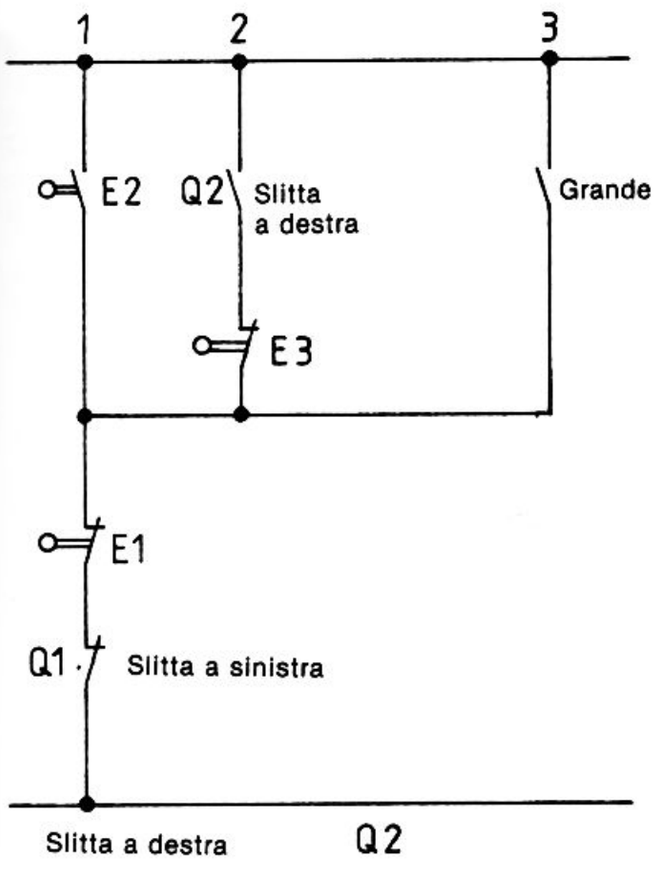


Fig. 8

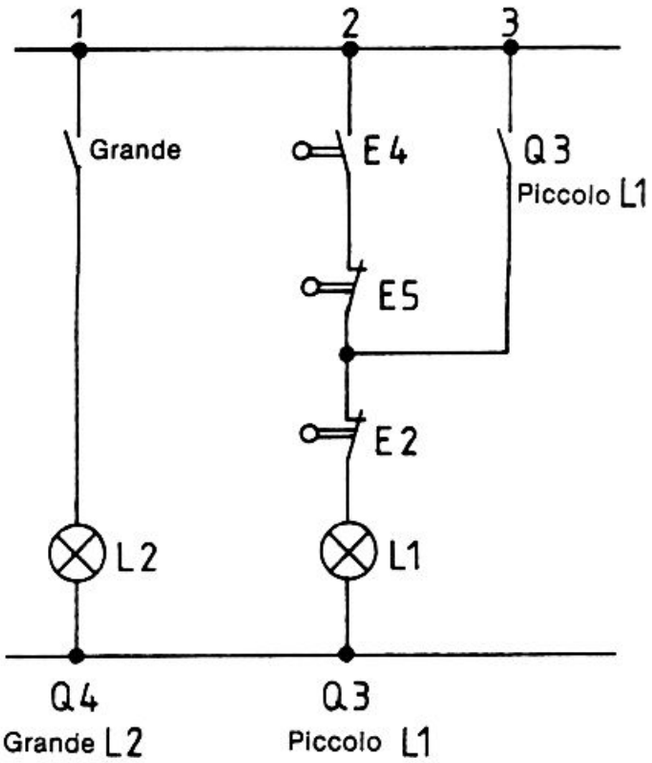


Fig. 9

quando verrà raggiunta di nuovo la posizione di partenza. La slitta rimane pertanto in questa posizione e deve ora spostarsi verso destra. Avviamo questo processo con il segnale E2 ed assegnamo una locazione per la memorizzazione, locazione che verrà cancellata quando la slitta sarà giunta nuovamente alla posizione E3.
La seconda funzione per "Slitta verso destra" viene generata attraverso la funzione "Grande". Con la selezione, il segnale viene memorizzato e cancellato nuovamente attraverso l'interruttore fine corsa E1. Perciò la slitta si sposta verso destra finché il segnale "Grande" è attivo. In linea di principio, il dispositivo di apertura E1 nel circuito di corrente non è indispensabile, dato che, come già detto, E1 cancella già la funzione "Grande". In pratica però, per la funzione, gli interruttori di delimitazione devono essere trovati direttamente. In questo caso si tratta dei circuiti di protezione del Motore M1.

Le lampade di segnalazione

Le lampade L1 ed L2 servono per l'indicazione del risultato della selezione. L1 si accende nella selezione "Piccolo" ed L2 nella selezione "Grande". Con la selezione "Grande" possiamo utilizzare direttamente il segnale della funzione "Grande" per accendere la lampada L2, che viene così automa-

ticamente spenta quando la slitta si allontana dall'interruttore E1. In questo momento si accenderebbe la lampada L1 indicando la selezione "Piccolo". Ciò naturalmente non è corretto e perciò dobbiamo o chiudere un occhio o generare un circuito di corrente per il riconoscimento della selezione "Piccolo". Scegliamo ovviamente la seconda soluzione ed usiamo il circuito seguente.
Una selezione "Piccolo L1" viene riconosciuta quando l'interruttore E4 è premuto mentre l'interruttore E5 non lo è. Viene quindi memorizzato il segnale per la lampada L1 che si mantiene attraverso il dispositivo di chiusura L1. Questo segnale verrà cancellato dopo il raggiungimento della posizione finale E2.

Il programma

Il Commodore è in grado, in base ai suoi operatori logici, di occuparsi direttamente dei circuiti di corrente. Osserviamo ora un esempio:
un semplice circuito di corrente deve portare la lampada all'accensione attraverso la pressione di

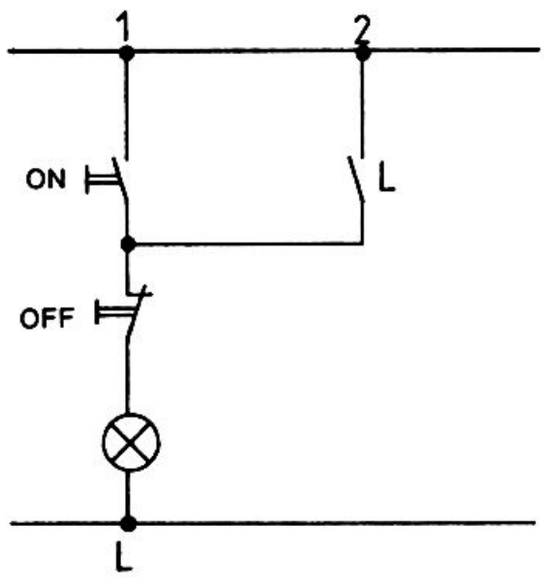


Fig. 10

un tasto. L1 rappresenta innanzitutto un memorizzatore che può essere attivato attraverso il secondo circuito OR. Premendo quindi il tasto AN, si attiva la memorizzazione su L1. Dopodiciò, indipendentemente dalla pressione del tasto ON, una corrente circola sugli interruttori L ed OFF verso la lampada. In tal modo abbiamo attivato la memorizzazione e possiamo spegnere la lampada mediante l'interruttore OFF.
Tutte le operazioni logiche vengono calcolate nel cosiddetto registro accumulatore del calcolatore.

Esempio: la funzione logica "ON" e "non OFF", corrispondente a lampada accesa, viene determinata dall'accumulatore. Consideriamo il listato seguente:

```
10 IF ON>0 AND OFF=0 OR L=1 AND OFF=0
THEN L1=1:GOTO 30
20 L1=0
30 .....
```

La struttura del programma del circuito di corrente "L" corrisponde proprio a entrambi i due circuiti di corrente. È perciò da notare che l'interruttore OFF è compreso sia nel primo che nel secondo circuito. Questo deve essere rappresentato anche nel programma in BASIC.

IF ON>0: se l'interruttore ON è premuto allora sull'ingresso della porta deve esserci un segnale 1. Se questo ingresso non arriva sulla porta PB0 ma sulla porta PB3, allora il valore di ingresso è uguale a 8. Per risparmiare l'istruzione IF PB3=8 THEN ON=1, possiamo direttamente chiedere se l'ingresso è >0.
Si continua con l'istruzione AND OFF=0: se l'ingresso dell'interruttore OFF è uguale a 0, allora il circuito di corrente 1 è chiuso e nell'accumulatore del calcolatore vi è giustamente uno stato logico 1.
Si prosegue con una istruzione OR che comincia con IF L1=1 AND OFF=0. Questo circuito di corrente è anch'esso chiuso quando L1 è premuto ed il tasto OFF non viene toccato. Nell'accumulatore viene fatto in tal modo un OR tra 1 ed il risultato già inserito. Il risultato finale dell'accumulatore è quindi nuovamente un 1. Dopo l'istruzione BASIC THEN L1=1 si prosegue con l'istruzione dopo THEN, ovvero alla variabile L1 viene assegnato il valore 1, dopo di che si salta alla linea 30.
Se nell'accumulatore non vi è alcun 1 allora non si prosegue con l'istruzione successiva al THEN ed il programma viene inviato alla linea 20 dove la variabile L1 viene azzerata.
In questa maniera è possibile riunire interi circuiti di corrente. Se il numero delle istruzioni OR fosse così elevato che l'editor non accettasse più le linee di programma, allora occorrerebbe riunire i circuiti relativi fino ad un punto comune. Alla fine poi questi circuiti separati saranno riuniti con un istruzione OR ed il risultato viene assegnato.
Nell'assegnazione occorre tenere presente che deve essere sempre assegnato o uno 0 oppure un 1.
Così il programma viene percorso circuito per circuito ed il risultato corrispondente viene assegnato a 1 oppure a 0.
Nel programma tutte le funzioni vengono realizzate secondo questa tecnica ed alla fine del programma vengono poi assegnate le variabili delle singole uscite. È importante che alla chiusura del programma ci sia un'istruzione di salto all'inizio del

programma stesso, il cui svolgimento deve essere ciclico. Tutte le funzioni di ingresso e di uscita vengono continuamente richieste ed elaborate. Per interrompere il programma useremo il tasto F1.

Il circuito

Per il comando della macchina selezionatrice, vengono usati in tutto 6 ingressi e 6 uscite. Poiché la User-port non ha a disposizione un numero di collegamenti così elevato, questi devono essere modificati di conseguenza. Con il circuito seguente, facile da comprendere, si può realizzare un economico ampliamento.
Il principio è il seguente: i terminali dati PB0-PB7 vengono alternativamente collegati in ingresso ed in uscita facendo sì che vi siano a disposizione 8 ingressi ed 8 uscite. Il nucleo centrale di questo circuito è il cosiddetto bus driver bidirezionale parallelo a 4 bit. Questo bus driver si occupa del collegamento alternato in ingresso ed uscita.

Modulo integrato: bus driver bidirezionale

Questo circuito ad alta integrazione, qui impiegato, ha un modo di funzionamento relativamente facile da comprendere. Illustriamo qui il concetto tecnico.

Bit: è un'abbreviazione per BINARY DIGIT. È la più piccola quantità rappresentabile in un sistema binario. In questo caso significa che i terminali dei dati possono assumere soltanto i valori 0 ed 1.

Bidirezionalità: gestione della direzione dei dati in due direzioni. Ovvero il componente può sia scrivere che leggere dati.

Bus: connettore a più fili, mediante il quale dei dati o dei segnali di controllo vengono trasferiti tra i componenti di un microprocessore. Abbiamo perciò un bus dati, un bus indirizzi ed un bus di controllo. In questo esempio si tratta di un bus dati.

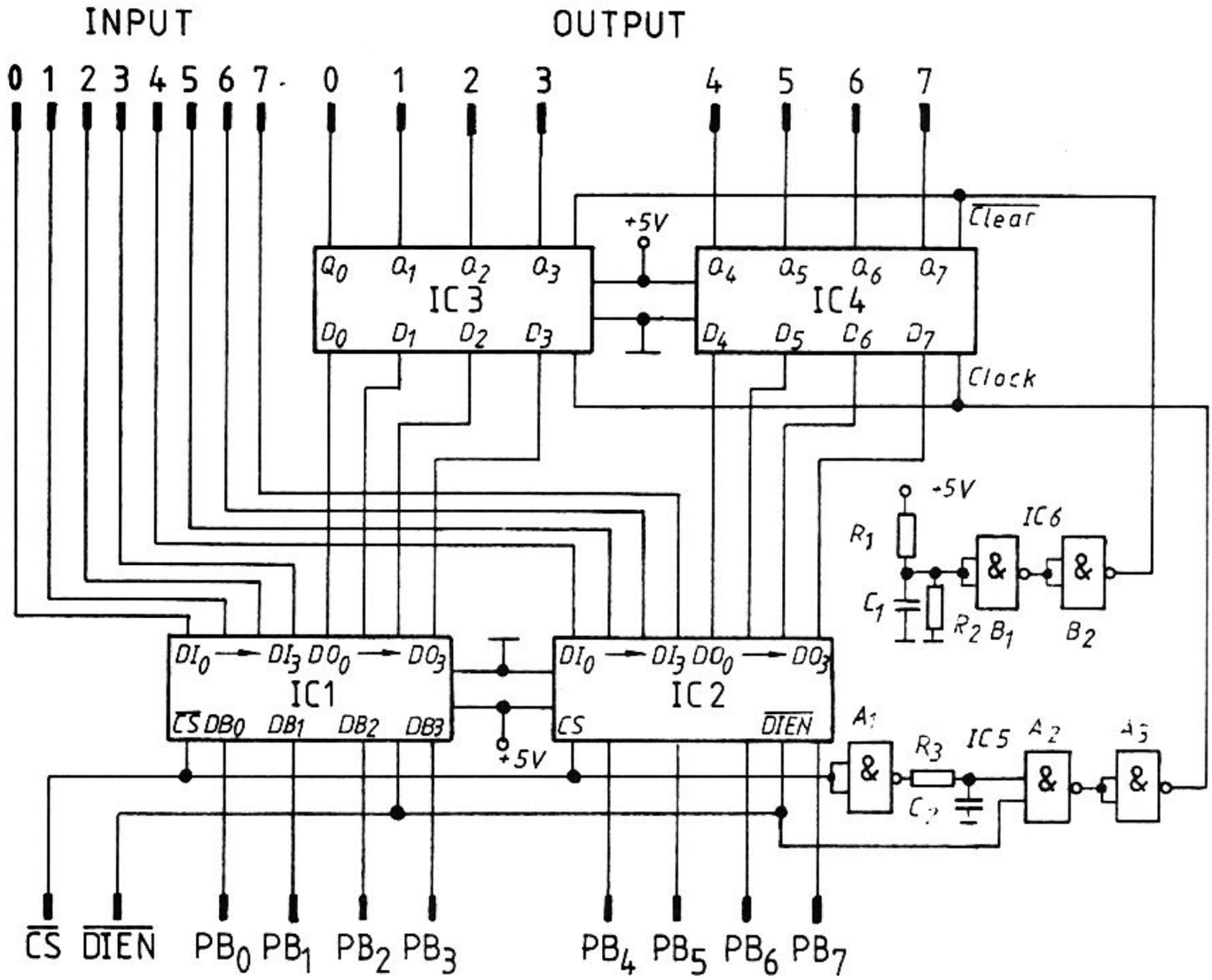
Driver: il componente driver ha la funzione di un amplificatore. Poiché l'uscita di un circuito interfaccia TTL a livello alto è in grado di fornire soltanto una corrente di pochi microampere, è conveniente inserire nel circuito un componente driver prima di inviare i segnali al bus dati. Il drive: ha un'impedenza di ingresso molto alta e non carica quindi il componente di uscita della User-port. Questo è in condizione di fornire una corrente di alcuni milliampere. I circuiti integrati IC1 e IC2 sono in grado di leggere o scrivere contemporaneamente 8 terminali di dati. L'ingresso dei dati viene ora trasferito dalla User-port al circuito integrato. Secondo lo stato di lettura o scrittura i dati vengono nuovamente collegati attraverso questo componente.

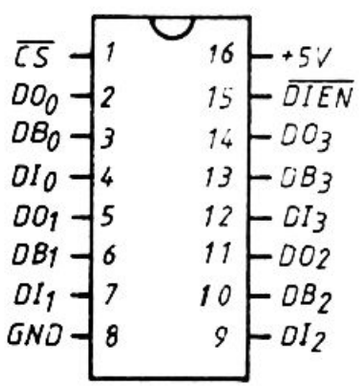
La determinazione della direzione

Come può essere determinata la direzione dei dati? Oltre ai pin dei dati, i circuiti integrati contengono altri due terminali di controllo. Il terminale di comando CS significa "Chip select". Se quindi è attivo questo terminale, il componente è in grado di lavorare. Il trattino sopra il simbolo CS significa che questo ingresso è attivo in condizione nulla. Ovvero, perché il componente sia attivo, occorre che sul pin CS vi sia una tensione nulla. Se non vi è una tensione nulla, allora i terminali dei dati dell'integrato presentano un'alta impedenza e non hanno perciò uno stato definito. Il secondo terminale di controllo è il pin DIEN (DIrection-ENable) che determina la direzione dei dati del componente. Per comprendere meglio il funzionamento considerate la figura sopra.
Cominciamo con il processo di lettura: devono essere letti i terminali DI0-DI3. Per leggere i dati sul pin di controllo DIEN deve esserci un segnale basso ed in questo modo viene attivato il corrispondente stadio del driver. Lo stadio rappresentato nella figura come T2 è ora in conduzione ed il flusso di dati va da DI0 verso DB0. Lo stadio T1 del driver non è attivato. Quando invece il terminale di controllo DIEN è allo stato logico 1, allora i dati passeranno da DB0 verso DI0 e lo stadio T2 non sarà attivo. Abbiamo così chiarito il funzionamento del controllo per l'ingresso e l'uscita dei dati.

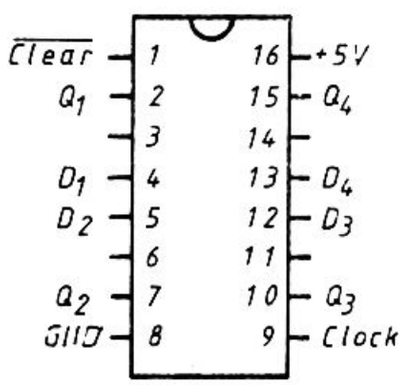
L'ordine

Dopo che abbiamo descritto la lettura e la scrittura del componente, occorre ora chiarire la sequenza dei segnali. Poiché in ogni flusso di dati si deve verificare una variazione del bus dati tra lettura e scrittura, questo bus deve essere inizialmente disattivato finché tutti i segnali sono presenti sul bus. Questo viene ottenuto attraverso il pin di controllo CS già visto, dopo di che determiniamo la direzione del flusso dei dati assegnando al pin DIEN lo stato logico 1 o 0. Dopo un breve tempo di attesa, i dati possono ora passare dal componente verso la User-port. Ciò significa che il segnale di controllo CS ha stato logico 0. Richiamiamo ancora una volta l'attenzione sul fatto che con uno stato logico 1 sul pin CS, tutti gli stadi del circuito sono disattivati.
Le funzioni per le elaborazioni dei dati sono perciò definite univocamente. I dati dall'esterno vengono letti sugli ingressi DI e trasferiti alla User-port, mentre i dati da scrivere vengono letti sulla User-port dagli ingressi DB0-DB7 e trasferiti all'esterno attraverso le uscite DO.
A questo livello il problema consiste nel fatto che noi dobbiamo continuamente alternare i collegamenti tra lettura e scrittura. Il nostro robot da tavolo non può riconoscere questo continuo "accen-





μPB 8216



SN74175

Fig. 11

dersi e spegnersi". In realtà vi è il rischio che, attraverso i contatti di scambio Ingresso/Uscita, i segnali esterni, siano motori, lampade e così via, vengano ad interferire nel processo di scambio. Ciò potrebbe provocare ad esempio il mancato funzionamento di un motore. Per garantire una corretta soluzione di questo problema, gli stati di uscita vengono di volta in volta memorizzati. Questa memorizzazione avviene mediante il circuito D-Flip-Flop. Come verrà più tardi descritto nel paragrafo "Il listato", esiste nella memoria di lavoro una cosiddetta configurazione di uscita che rappresenta esattamente i processi di uscita. Gli stati di uscita vengono quindi salvati internamente dopo ogni variazione logica e, dopo la memorizzazione dell'uscita, nuovamente assegnati al D-Flip-Flop.

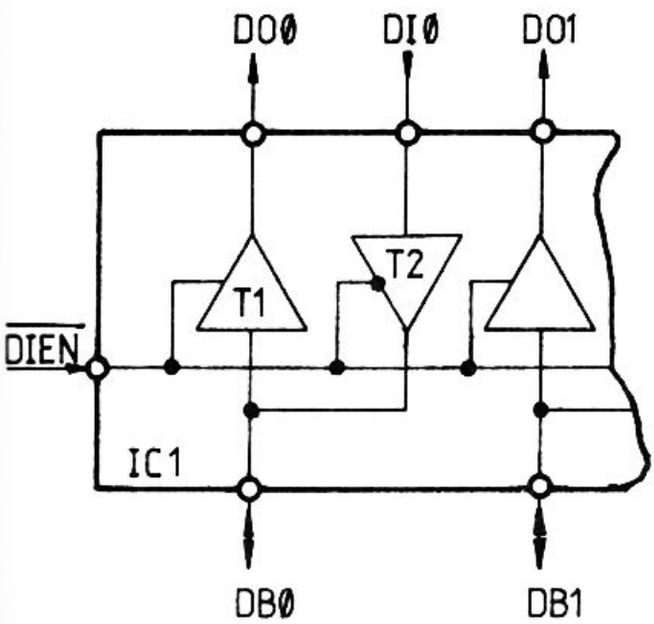


Fig. 12

Il registro

Il D-Flip-flop qui impiegato è un registro D a 4 bit. Questa è una forma alternativa del Flip-flop bistabile che può avere sulla sua uscita Q due stati:

Uscita Q = 0 Volt e quindi stato logico 0
Uscita Q = 5 Volt e quindi stato logico 1

Ciascuno dei due stati logici di uscita è stabile e resta costante fintantoché dall'esterno non giunge un impulso di trigger di cambiamento di stato.

Isolamento delle uscite

Tutti gli stati del processo vengono definiti con precisione dal programma. Poiché in realtà, all'accensione dell'apparecchio ovvero dell'interfaccia, al D-Flip-flop giunge uno stato non definito, nel processo può essere generato un errore. A questo scopo il D-Flip-flop contiene un ingresso di reset. Nel disegno, questo ingresso è indicato con il simbolo CLR. Poiché sulle uscite Q0-Q3 sono collegati dei motori, delle lampade spia e così via, dobbiamo accertarci che venga sempre generato uno stato definito. Durante l'elaborazione il programma si occupa di questo compito, ma cosa accade quando il programma non è ancora iniziato? Appena il circuito è alimentato dalla tensione di alimentazione, l'uscita del Flip-flop assume uno stato qualunque che può essere o lo stato logico 0 oppure lo stato logico 1. In questo modo i motori potrebbero ad esempio incominciare a girare e così via. Affinché ciò non si verifichi, abbiamo modificato il nostro circuito con un reset automatico B1 e B2. Come funziona questo circuito aggiuntivo? I circuiti integrati qui utilizzati sono costituiti da trigger di Schmitt NAND con due ingressi ciascuno. Siccome entrambi gli ingressi sono collegati, il segnale viene soltanto invertito. Il componente ha una soglia definita di circa 1,6 Volt, e quindi quando la tensione d'ingresso è inferiore a 1,6 Volt, sull'uscita abbiamo un segnale logico 1 (inversione). Quando la soglia viene oltrepassata, allora l'uscita passa allo stato logico 0. Quando la scheda viene alimentata, il condensatore C1 si carica lentamente attraverso la resistenza R1. Questo condensatore impiega circa un secondo prima di caricarsi ad una tensione superiore a 1,6 Volt. Per questo tempo l'ingresso Clear del D-Flip-flop viene quindi mantenuto allo stato logico 0 e si garantisce perciò uno stato di uscita ben definito. Il secondo circuito aggiuntivo serve per la generazione del segnale di clock. I segnali per DIEN e CS vengono inviati dall'ingresso clock del D-Flip-flop collegati logicamente e ritardati nel tempo. In questo modo i dati giacenti sul D-Flip flop vengono automaticamente trasferiti non appena CS diventa nullo ed il condensatore C2 è carico.

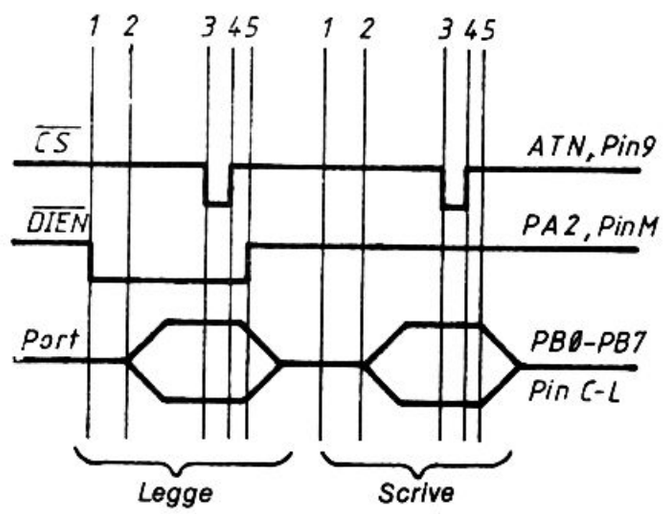


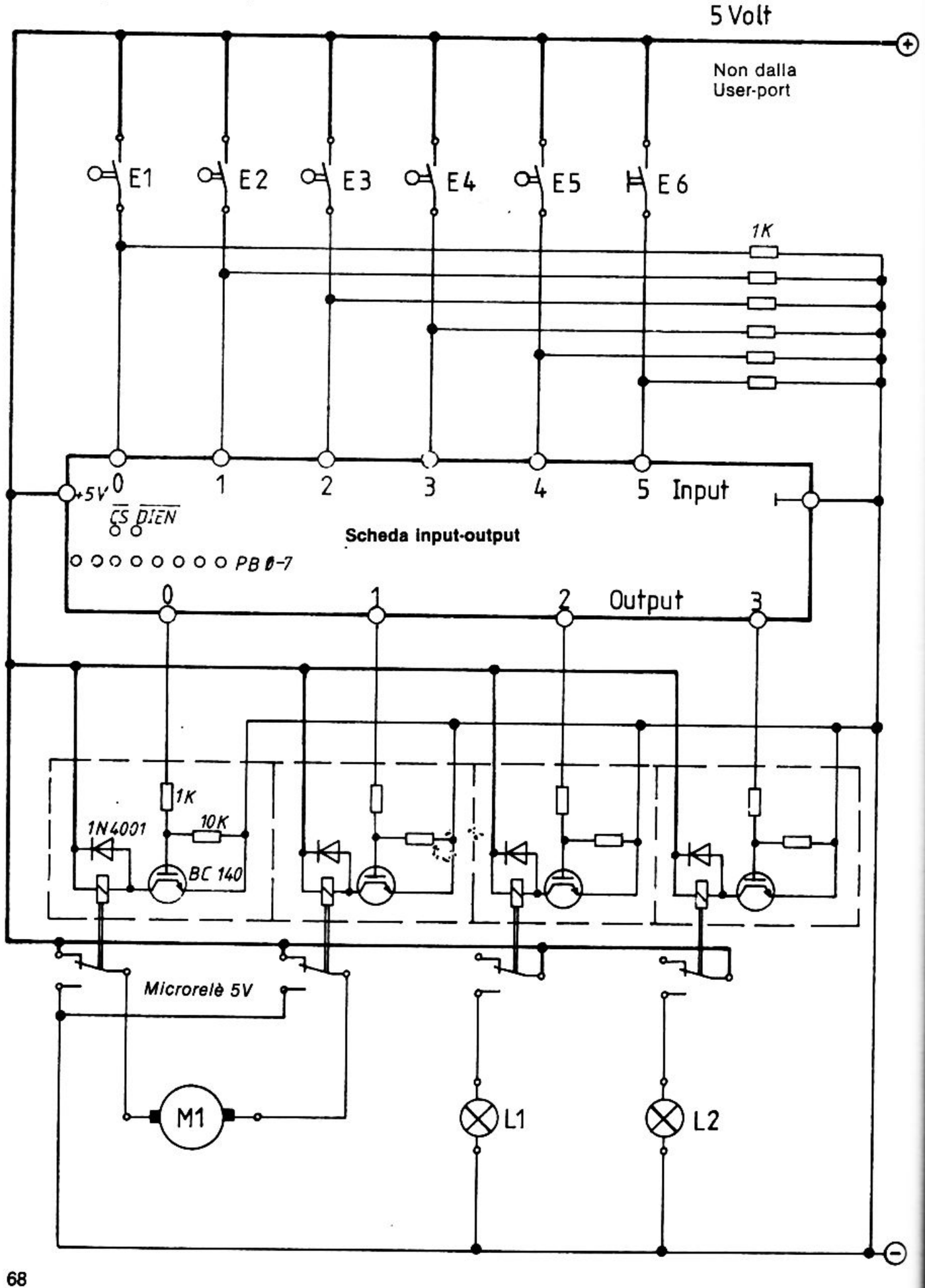
Fig. 13

Schema dei collegamenti al robot da tavolo

Fig. 14

Usare alimentazioni separate

Non collegare mai la user-port alla tensione



Il listato

Il programma per il comando della macchina selezionatrice è codificato in BASIC. I passi del programma sono stati ridotti al minimo indispensabile e nessun commento REM è stato inserito all'interno delle istruzioni. Poiché la slitta si muove in modo relativamente lento ed una camma più grande potrebbe facilmente rompersi, la durata del segnale è sufficiente, ed il programma scritto in BASIC può assolvere le rispettive funzioni. Per un compito più veloce, in realtà, questo programma non è appropriato e bisogna scrivere un programma in Assembler.

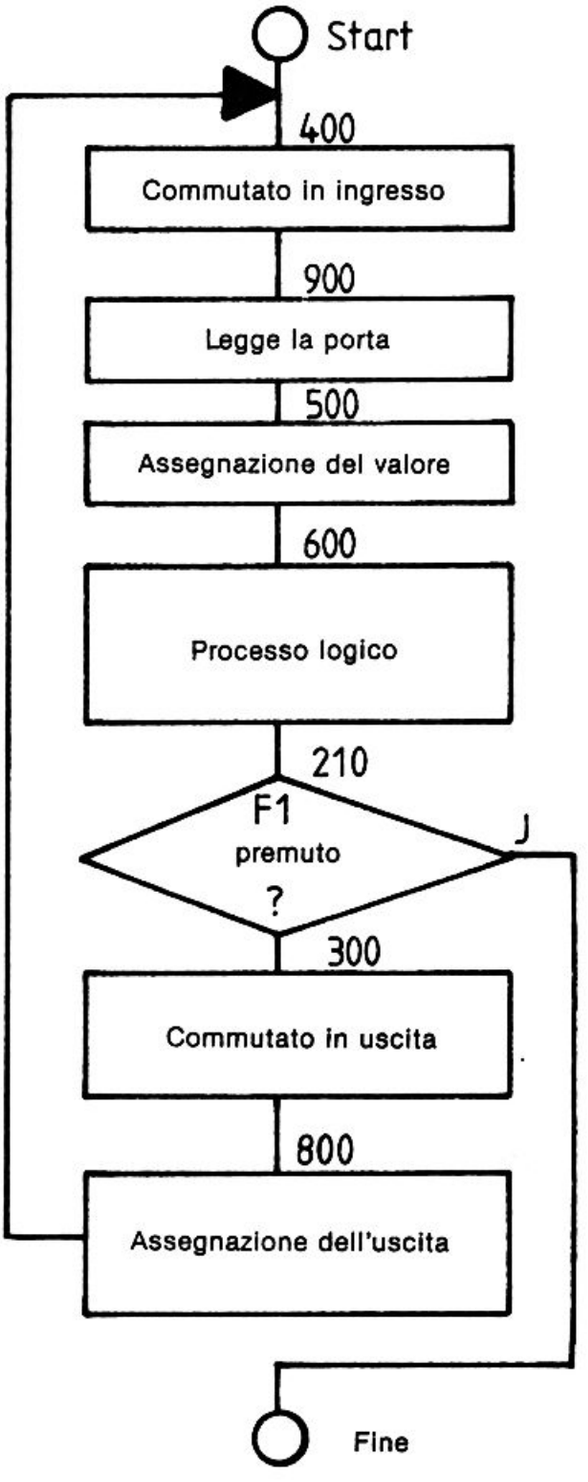


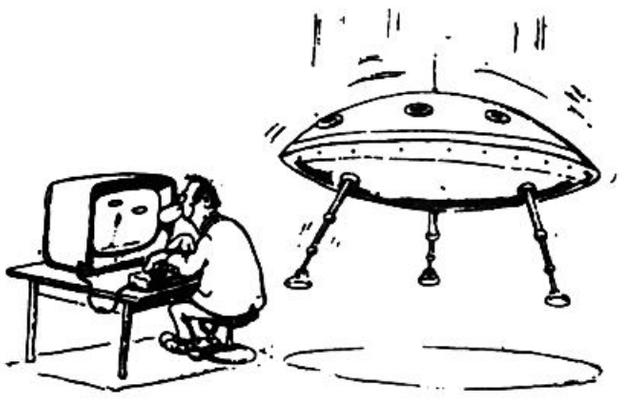
Fig. 15

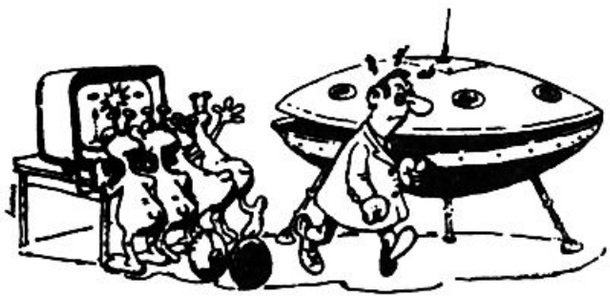
Dapprima vengono rappresentate le variabili nelle linee REM. Nella linea 200 inizia il programma vero e proprio dal quale si viene inviati alle singole subroutine. Cominciamo con la subroutine 400: questa subroutine si occupa della selezione come ingressi delle connessioni della porta. All'interno di questa subroutine si viene inviati alla subroutine 900 che legge i dati presenti sulla porta, attraverso la variabile X. Come si vede nel diagramma di flusso, viene dapprima azzerato il pin DIEN ed in tal modo, come descritto nel manuale di istruzioni, i dati vengono letti dall'esterno. Questo azzeramento avviene mediante il comando POKE PO+0,147+8. Dopo di ciò il contenuto della User-port viene assegnato alla variabile X. Ciò è indispensabile in quanto il componente dell'interfaccia dopo la lettura assume un'impedenza alta ed allora tutte le uscite della User-port vengono automaticamente messe a 1. Questo comportamento è specifico del Commodore. Affinché questi valori non vengano letti, durante il processo di lettura della User-port, viene assegnata la variabile X. Dopo di che il pin CS viene messo a 1 con l'istruzione POKE PO+0,147.

Dopo la lettura dei dati la subroutine alla linea 500 può assegnare ai singoli ingressi i valori assegnati alla variabile X mediante una semplice istruzione FOR. In tal modo vengono determinate le variabili d'ingresso.

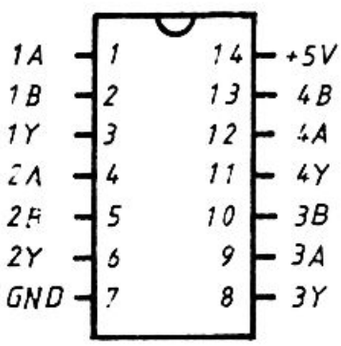
Segue quindi nella subroutine 600 la realizzazione della rispettiva logica. Questa logica rappresenta precisamente i circuiti di corrente già descritti nel testo. Osservandoli, qualunque utente di BASIC comprende chiaramente che il Commodore impiega molto tempo per l'elaborazione delle istruzioni. Questa parte del programma BASIC sarebbe preferibile che fosse codificata in Assembler. Dopo la determinazione delle variabili di uscita Q1-Q4, l'uscita della porta viene connessa in uscita dalla subroutine 800. Segue poi un salto alla subroutine 300 che seleziona le porte come uscite. Con il comando POKE PO+0,151 il terminale DIEN viene messo a 1 ed in tal modo l'interfaccia sa che i dati della User-port vengono scritti. Con l'istruzione POKE PO+3,255 la porta viene selezionata come uscita e, nella subroutine 800, si ha la scrittura dei dati. Viene poi verificato se l'uscita della User-port sta effettivamente scrivendo. Le variabili Q1-Q4 vengono sommate ed assegnate all'uscita B della porta. I dati adesso sono sul bus e l'azzeramento del pin CS provoca il trasferimento dei dati nel D-Flip-flop.

Dopo che il trasferimento è avvenuto e che il pin



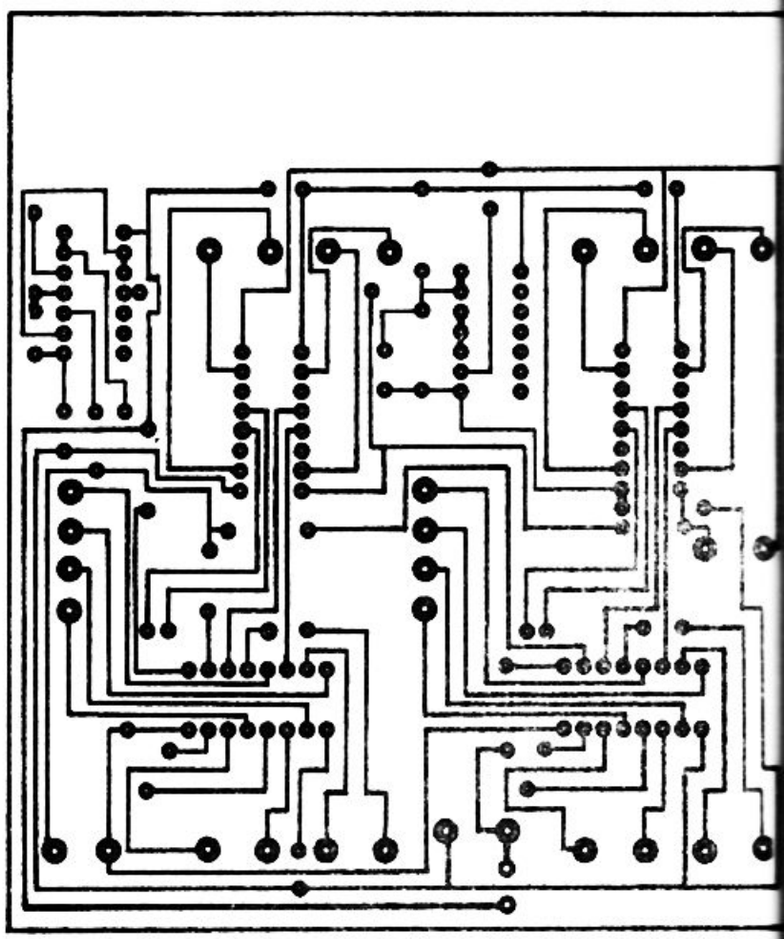


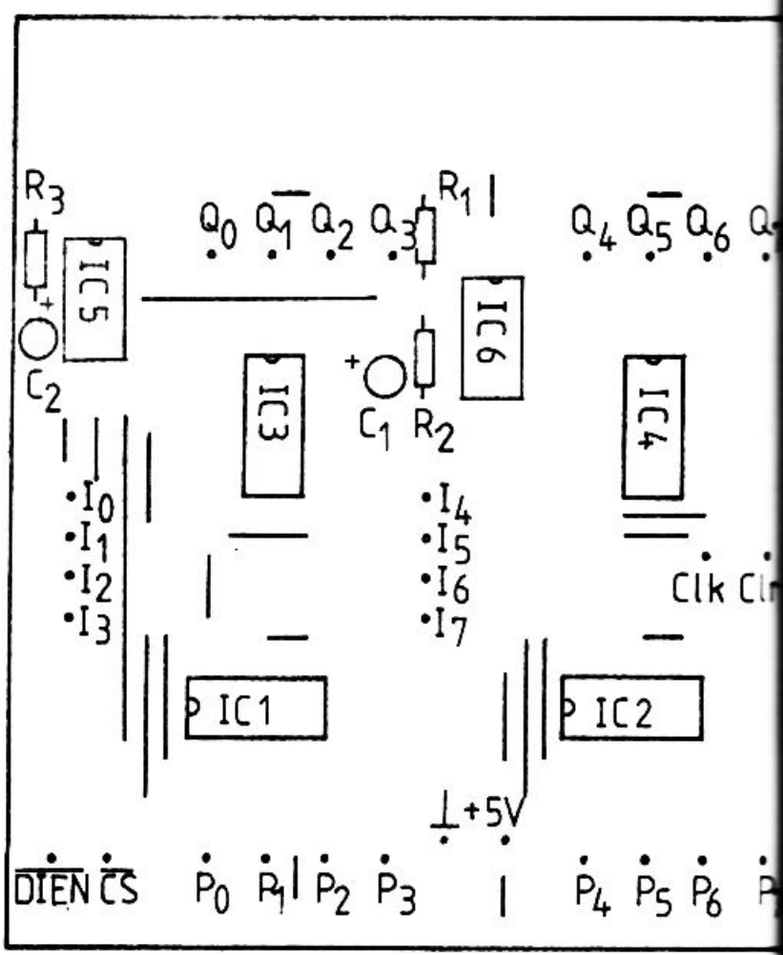
Disegni: Daniel Le Noury, Sybex



SN 74132

Fig. 16





CS è stato posto nuovamente a 1, può ripetersi il ciclo generale. Gli indirizzi per i pin DIEN e CS sono:

PA2 = 56576 bit 2 = DIEN
ATN = 56576 bit 3 = CS

Elenco dei componenti:

R1 = 100 KOhm
R2 = 39 KOhm
R3 = 100 Ohm
C1 = 3,3 nF
C2 = 68 microF
IC5,6 = SN74132
IC1,2 = 8216
IC3,4 = 4175

Collegamenti esterni:

6 resistenze da 1 KOhm
4 interruttori di potenza
1 Alimentatore 5 Volt 1 Ampere
1 scatola di montaggio Fisher-Technik-Computing
1 buona fortuna.

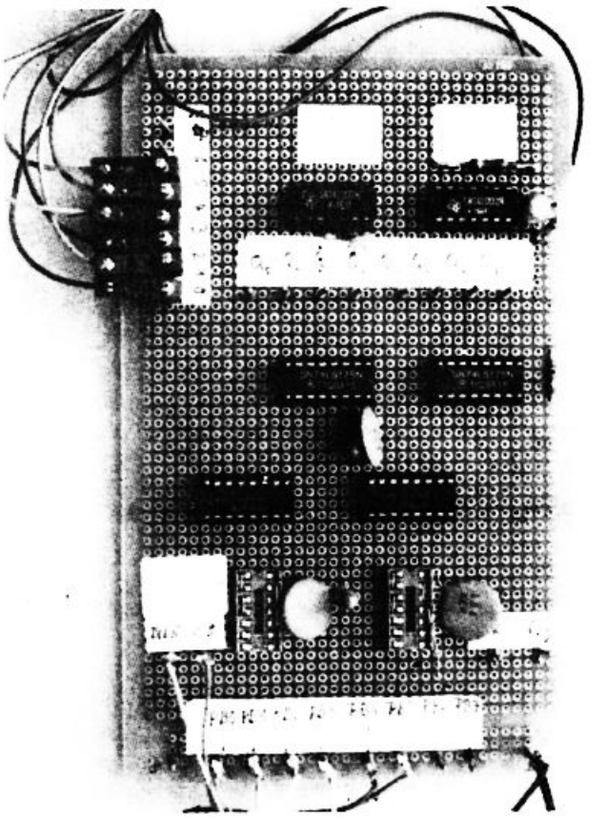


Fig. 17

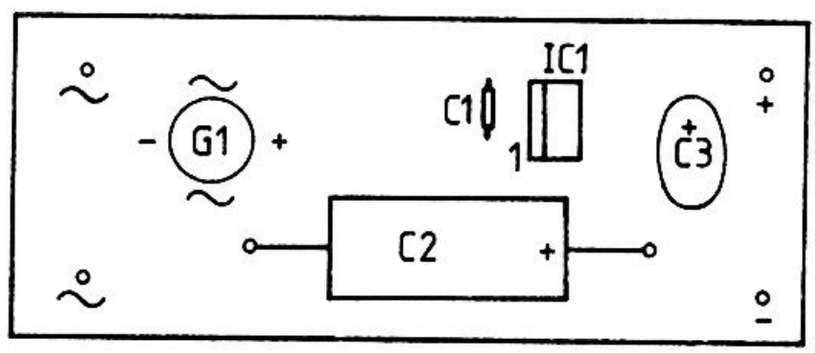
```
5 REM C64 COME ROBOT DA TAVOLO
10 PO=56576
20 PRINTCHR$(147);CHR$(144);"START!          STOP = F1 "
40 REM E1=SLITTA DESTRA
50 REM E2=SLITTA SINISTRA
60 REM E3=SLITTA IN POSIZIONE DI RIPOSO
70 REM E4=SELEZIONE-GRANDE-PICCOLO
80 REM E5=SELEZIONE=GRANDE=PICCOLO
90 REM E6=INIZIO ORDINAMENTO
100 REM Q1=MOTORE SINISTRO
110 REM Q2=MOTORE DESTRO
120 REM Q3=LUCE1=PICCOLO
130 REM Q4=LUCE2=GRANDE
140 REM DIEN=PA2=PO+0 BIT2
150 REM CS=ATN=PO+0 BIT3
160 REM GR=GRANDE  PC=PICCOLO
170 REM GS=POSIZIONE DI RIPOSO SS=INIZIO ORDINAMENTO
180 :
190 :
200 GOSUB 400:GOSUB 500:GOSUB 600:GOSUB 800
210 GETX$:IFX$=CHR$(133) THEN Q1=0:Q2=0:Q3=0:Q4=0:GOSUB 800:GOSUB400:END
220 GOTO 200
230 :
240 :
300 REM COLLEGAMENTO USCITA
310 POKE PO+0,151:REM DIEN=1
320 POKEPO+3,255:REM PORTA B COME USCITA
330 RETURN
```

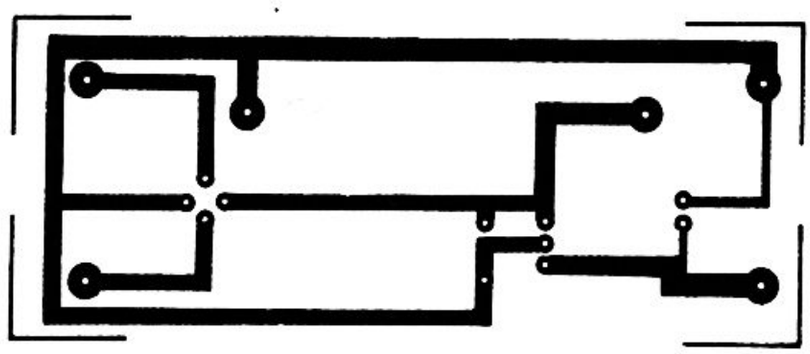
Come aumentare la tensione del C64

Sono stati descritti molti circuiti elettronici i quali hanno tutti bisogno di un'alimentazione. Nella maggior parte dei casi questa può essere ricavata direttamente dalla User-port, mentre per alcuni di questi circuiti è richiesta una corrente che costituisce un carico troppo elevato per la User-port e "tira il collo" alla sicurezza del C64.
Perché ciò non si verifichi, occorre usare per l'alimentazione l'alimentatore descritto di seguito. Tutti i circuiti che non possono essere alimentati direttamente dalla User-port, possono fare uso di questo alimentatore. Se siete in dubbio riguardo a qualche circuito, dovete cercare nel testo corrispondente in modo che possiate provare il circuito con fiducia.
Il nucleo centrale di questo alimentatore è un circuito integrato per la regolazione della tensione, protetto contro il cortocircuito. La sua tensione di uscita è altamente costante. Oltre a questo regolatore di tensione sono necessari soltanto pochi componenti passivi che servono per la sicurezza del circuito e per la sua affidabilità.
Il regolatore di tensione deve essere montato su un radiatore in modo da poter disperdere il calore. Per la sicurezza generale è meglio chiudere l'intero alimentatore in una scatola e l'uscita della tensione può avvenire mediante due connettori a banana.

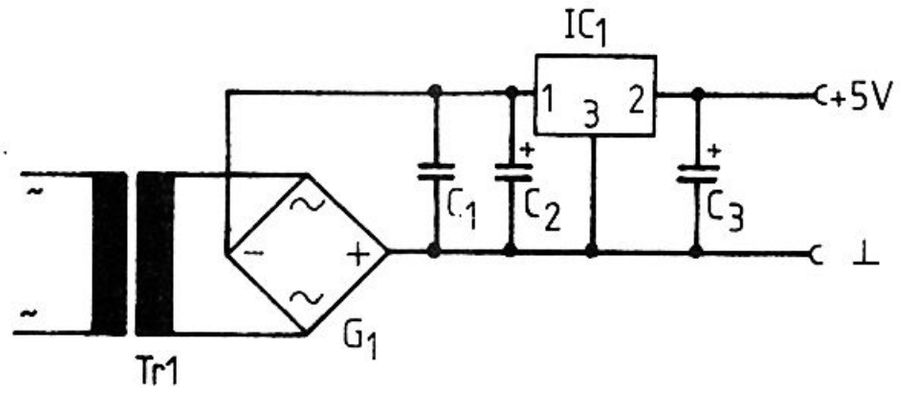
Elenco dei componenti dell'alimentatore:

Tr1 = Trasformatore 7-12 Volt AC, 1-1,5 A
G1 = ponte raddrizzatore B40C1500
C1 = 0,1 microF
C2 = 2200 microF
C3 = 4,7 microF
IC1 = regolatore di tensione 7805.

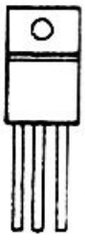




Alimentatore



Denominazione
dei terminali di IC1



132

Fig. 2

Come modificare il mio C64

Chi vuole modificare il suo C64, per esempio per trasformarlo in un robot, ha bisogno di diversi aiuti. Questo piccolo manuale per gli amanti del "Fai da te" può perciò risultare notevolmente utile. I circuiti descritti sono stati realizzati tenendo in considerazione i seguenti aspetti:

— I circuiti devono essere facili da costruire e da comprendere anche per un profano.

— I componenti devono essere economici e facili da reperire.

Costruzione dei circuiti

Vi sono due possibilità per realizzare i circuiti indicati.
1. Chi non ha alcuna esperienza con i componenti elettronici dovrebbe realizzare i circuiti su una piastra sperimentale, che può essere acquistata in qualsiasi buon negozio specializzato o presso i rivenditori di componenti elettronici per corrispondenza. Il vantaggio di queste piastre sta nel fatto che possono essere sempre utilizzate per circuiti diversi. Su di una piastra in materiale sintetico, sono riportate le piste conduttrici. Nei contatti della griglia, possono essere con facilità inseriti i componenti che vengono mantenuti nella sede da un contatto a molla. Un circuito può perciò essere assemblato su una di queste piastre, senza far uso del saldatore.
2. Se un circuito non deve essere realizzato a livello sperimentale su una piastra, ma deve essere già saldato sulla piastra definitiva, allora si può usare a questo scopo una piastra che dispone di una griglia su cui saldare i componenti. Il materiale fondamentale consiste di un materiale sintetico isolante sulla quale si trovano molte piazzuole rotonde di rame che possono essere saldate. I componenti possono essere inseriti nella piastra forata dalla parte non ramata (lato componenti). Dopo la saldatura occorre tagliare i connettori a circa 2 mm dalla piastra. I collegamenti possono essere effettuati mediante un unico filo di rame argentato da 0,5 mm oppure con un filo isolato.
I circuiti possono così essere realizzati seguendo il progetto del circuito. I numeri sui componenti rappresentano i contrassegni dei collegamenti.
Il prossimo passo sarebbe già una costruzione professionale su un circuito stampato. La realizzazione di un circuito del genere non viene descritta. Per il dilettante vi è la possibilità di acquistare il necessario per la realizzazione o addirittura il circuito già fatto. Questa è la maniera più facile e più veloce per provare i circuiti corrispondenti sul calcolatore.

Disposizione dei componenti

I connettori dei componenti non devono essere tagliati troppo vicino al componente altrimenti l'isolante può raggiungere la molla di contatto. La lunghezza dei connettori va regolata in modo che la distanza tra i componenti e le piegature non sia inferiore a 2 mm. Per piegare i connettori è meglio usare una pinzetta.

Riconoscimento delle resistenze

Poiché i componenti diventano sempre più piccoli, non è più possibile trovare, sul corpo del componente stesso, lo spazio sufficiente per stampare tutte le informazioni utili. Perciò è necessario avere un codice di riconoscimento che, nel caso delle resistenze, è costituito da quattro strisce colorate. Le prime due strisce indicano le prime due cifre del valore della resistenza. Il terzo anello è il cosiddetto moltiplicatore il cui valore va moltiplicato per il valore indicato dalle prime due strisce. Il quarto colore indica la tolleranza del valore della resistenza che dipende dal procedimento di fabbricazione del componente.
Osservate la seguente tabella delle resistenze ed il relativo esempio di lettura del valore di una resistenza in base al codice dei colori.

Una corretta saldatura

Usate un saldatore con temperatura di circa 260 °C per ottenere un'ottima connessione saldata. Il dilettante commette spesso l'errore di restare troppo tempo su una saldatura con un saldatore troppo caldo cosa che può portare ad un surriscaldamento del componente. È quindi opportuno usare un saldatore con potenza nominale non superiore a 30 W. Ed ora passiamo al procedimento di saldatura vero e proprio. Mettete lo stagno e la punta del saldatore contemporaneamente sul materiale da saldare. Lo stagno quindi fonde e si lega con il connettore, spostate un po' il saldatore ed allontanatelo poi dal terminale saldato. La punta del saldatore deve restare a contatto del conduttore al massimo per circa 3 secondi. Tempi superiori possono portare alla distruzione del componente se la temperatura della saldatura non è controllata mediante una pinzetta che disperda il calore.
Il rame, e specialmente i sottili terminali di collegamento, sono particolarmente sensibili alla corrosione. Si usa perciò, nei circuiti elettronici, unicamente il colofonio come disossidante. Usando stagno con anima disossidante (soprattutto stagno-colofonio), si evitano tutti i problemi connessi con la scelta dello stagno e del disossidante.
Le piste ramate devono essere pulite dallo sporco e dal grasso. Ciò può essere fatto strofinandole con uno straccio imbevuto di alcool. Per i circuiti integrati, è conveniente utilizzare gli zoccolini poiché la saldatura di un circuito integrato sulla piastra ramata può portare il circuito integrato ad alta temperatura. Inoltre in questo modo è possibile sostituire facilmente il circuito integrato stesso se difettoso.

Gli attrezzi da lavoro

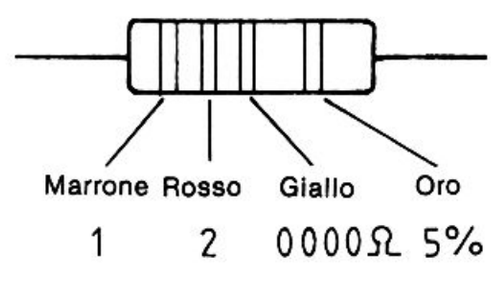
Per la corretta realizzazione di un circuito servono alcuni determinati strumenti fondamentali, la maggior parte dei quali esiste sicuramente in ogni casa:

* 1 piccola lametta
* 1 tronchesino appuntito
* 1 scatola di cacciaviti
* 1 pinzetta
* 1 trapano con punte da 1,3 a 1,5 mm
* 1 saldatore a punta sottile

Un multimetro è assolutamente indispensabile per la realizzazione di questi circuiti elettronici e deve potere essere utilizzato sia come voltmetro che come amperometro. I campi di misura variano da 1 Volt a 15 Volt e da 1 mA a circa 1 A.
Considerando con cura i punti sopraindicati, andrà tutto bene. Al lavoro e buon divertimento!

Colore	1 Anello = 1 cifra	2 Anello = 2 cifra	3 Anello = Moltiplicat.	4 Anello = Tolleranza
Nero	—	0	10^0	20%
Marrone	1	1	10^1	1%
Rosso	2	2	10^2	2%
Arancio	3	3	10^3	
Giallo	4	4	10^4	
Verde	5	5	10^5	
Blu	6	6	10^6	
Viola	7	7	10^7	
Grigio	8	8	10^8	
Bianco	9	9	10^9	
Oro	—	—	10^{-1}	5%
Argento	—	—	10^{-2}	10%

Fig. 1



Introduzione alla computer-robotica

Per l'utilizzazione della User-port, le normali conoscenze della maggior parte dei programmatori in BASIC, non sono sufficienti. Qui gioca sicuramente un ruolo importante il modo e la maniera di utilizzare la User-port. Cosi, mentre si può imparare facilmente il semplice comando per la connessione della porta in ingresso ed in uscita, è in realtà più difficile usare contemporaneamente una sola porta come ingresso e come uscita. Il comando esterno della User-port da parte di barriere luminose, interruttori o altri sensori, deve produrre un determinato effetto in un programma. Un altro impiego conveniente è ad esempio la realizzazione di un allarme antifurto.

Simulazione sullo schermo

Lo stato logico della User-port può essere rappresentato per mezzo di più diodi luminosi usando il circuito di prova della User-port descritto al capitolo "Il C64 come gioco di luce".
Per chi non ha a disposizione questo circuito di prova, occorre dapprima illustrare un programma che simula sullo schermo il funzionamento della User-port. Avrete quindi la possibilità di usare un programma di prova. Inserite il programma BASIC riportato nel listato dalla linea 1000 alla linea 2050. Il nucleo fondamentale del simulatore è costituito da un registro ad 8 bit che viene indicato sullo schermo. Questo registro ad 8 bit rappresenta metaforicamente la User-port e vale sia per la porta A che per la porta B. L'indirizzo di base del componente di uscita CIA è 56576. Il registro A ha indirizzo 56576+0 mentre il registro B ha indirizzo 56576+1. Nel seguito indicheremo l'indirizzo di base con la variabile PO. Osservate ora la figura della parte hardware della User-port (figura della scheda). Vi sono a disposizione per la porta le uscite da PB0=Pin C a PB7=Pin L. Questi 8 segnali rappresentano sempre l'immagine del registro PO+1. Attraverso il programma di simulazione, avete ora la possibilità di modificare i contenuti dei registri PO+1 e PO+3. Considerate dapprima come dato il listato del programma di simulazione visto che non siete ancora in grado di comprenderlo.

Semplici sciocchezze

La User-port è bidirezionale ovvero può trasferire dati dall'esterno al calcolatore oppure dal calcolatore verso l'esterno. La decisione se leggere dati dall'esterno verso il calcolatore, o scriverli in direzione opposta, è contenuta nel registro PO+2 per la porta A e PO+3 per la porta B. Dopo che avete introdotto il programma, salvatelo sulla memoria di massa ed avviatelo con il comando RUN. Sullo schermo comparirà l'immagine del registro per la porta B. L'uso della porta A non è possibile in quanto questa non è collegata per via hardware alla User-port. Le nostre considerazioni saranno perciò sempre limitate al registro B con l'indirizzo PO+1. L'influenza sulla direzione dei dati si ottiene, come già menzionato, con il registro PO+3 per la porta B. Inserite ora il seguente programma:

```
10 PO=56576
20 POKE PO+3,0
30 GOSUB 1000
40 FOR K=1 TO 2000:NEXT
50 POKE PO+3,255
60 GOSUB 1000
70 FOR K=1 TO 2000:NEXT
80 GOTO 20
```

Avviate il programma con il comanndo RUN e considerate attentamente il contenuto del registro PO+3 e la direzione delle frecce.

Nota

La direzione dei dati del registro B può essere selezionata in ingresso ovvero in uscita mediante il registro dati PO+3=56576+3. Se il registro è a 0, la porta funziona come ingresso, mentre se il registro è a 1 funziona in uscita. L'istruzione POKE 255 genera un 1 in tutti i bit mentre POKE 0 provoca l'azzeramento di tutti i bit.

Descrizione del programma

Iniziamo con la descrizione del programma dalla linea 10 alla linea 80. Come prima cosa, viene assegnato l'indirizzo di base PO=56576. Questa istruzione deve essere assolutamente presa in considerazione da una subroutine dell'utente. Nella linea 20 viene selezionata la direzione dei dati in ingresso e si salta poi alla subroutine 1000 nella quale il nostro simulatore compare sullo schermo. Considerate la direzione dei dati ed il relativo registro. Nella linea 40 si ha a questo scopo un ciclo di attesa di circa 2 secondi al quale segue, nella linea 50,

la selezione della direzione in uscita. Nella linea 60, dopo il salto alla subroutine, viene modificata l'immagine sullo schermo. Notate la variazione di direzione ed il corrispondente contenuto del registro. Questo processo si ripete fintantoché non viene premuto il tasto RUN/STOP.
Ed ora veniamo al simulatore. Nella riga 1010 si ha la domanda se l'immagine della User-port è già mostrata sullo schermo o no. In questo caso si salta alla linea 1230 e si inizia con l'immagine del contenuto dei registri. In caso contrario invece l'immagine del simulatore della User-port viene generata sullo schermo per mezzo delle istruzioni PRINT corrispondenti. Non è necessaria alcun'altra spiegazione poiché questo è un puro e semplice comando PRINT.
All'interno della subroutine 1000 si trova la subroutine 2000 che inizia la trasformazione del registro in rappresentazione binaria. Questo avviene ad esempio per l'immagine delle uscite della User-port nella linea 1230. Lo stato della variabile Z viene riassegnato e si salta quindi alla linea 2000 e il numero Z viene trasformato in una stringa corrispondente. Il risultato si trova nuovamente nella stringa RE$. Nella linea 1240 si apre un ciclo FOR che, mediante il comando POKE, riempie il contenuto dello schermo con il corrispondente numero di RE$. La variabile X è necessaria, poiché tra i comandi POKE viene fatto un ciclo di attesa. Questo processo si ripete poi all'indirizzo 1300 per la rappresentazione della direzione dei dati. Il corrispondente codice numerico della direzione, viene assegnato nelle linee 1312 e 1314. Alla linea 1360, viene finalmente assegnato alla variabile Z il contenuto del registro direzionale PO+3. Dopo di che si ha, come al solito, il salto alla subroutine 2000 e l'indicazione del risultato sullo schermo.
In questo modo si è provveduto alla rappresentazione sullo schermo dello stato della User-port, della direzione dei dati e del contenuto del registro direzionale. L'utente può provare questa subroutine a volontà e vedere lo stato dei registri sullo schermo.

Elaborazione a singolo bit

Dapprima ancora un esempio: il programma di esempio dalla linea 10 alla linea 80 resta invariato. Occorre solo cambiare leggermente le linee 20 e 50 in:

```
20 POKE PO+3,1
50 POKE PO+3,0
```

Introducete le modifiche ed avviate il programma con RUN. Vedrete che sullo schermo il bit a destra cambia di stato e cambia la direzione dei dati. Da qui la considerazione: nella linea 20 inseriamo un 1 nel registro direzione dei dati PO+3 con un comando POKE. Allora nel registro PO+3 vi è il valore 1 e sullo schermo la rappresentazione sarà 00000001. Dopo il tempo di attesa con il ciclo FOR, nella linea 50 il registro direzione dei dati viene nuovamente azzerato e sullo schermo compaiono allora 8 zeri. Contemporaneamente viene cambiato il simbolo corrispondente alla direzione dei dati al di sopra del registro direzionale. Mentre il nostro registro direzionale viene commutato tra ingresso ed uscita, tutti gli altri registri restano costantemente come ingressi. Com'è facile da capire, ci si può rivolgere ad ogni singolo bit del registro. Se un singolo bit del registro direzionale è uguale a 1, allora la sua direzione dei dati è commutata in uscita e sulla User-port compare uno 0. Se il bit del registro direzionale è a 0, lo stesso accade per la corrispondente direzione dei dati nel computer e sul registro della User-port compare un 1.
Scrivete ora un valore qualsiasi nelle linee 20 e 50 ed osservate attentamente lo schermo. Non potete usare nessun valore minore di 0 o maggiore di 255. Introducete qualsiasi numero decimale compreso tra 0 e 255 nel registro corrispondente. Se volete ad esempio collegare come uscite i primi quattro terminali della User-port e come ingressi i restanti quattro, allora occorre introdurre nel registro corrispondente il valore appropriato. Considerate perciò la seguente figura.

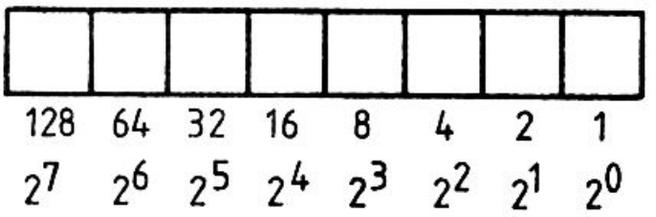


Fig. 1

Ogni bit all'interno di un registro ha un determinato valore. Nel registro vengono contati da destra verso sinistra. Il bit minore è quindi il bit nella posizione più a destra e vale 2^0. Il bit più significativo è invece il bit di sinistra che ha valore 2^7. Questi valori vengono sommati quando nel bit corrispondente vi è un 1. Perciò la configurazione seguente ha il valore sottostante:

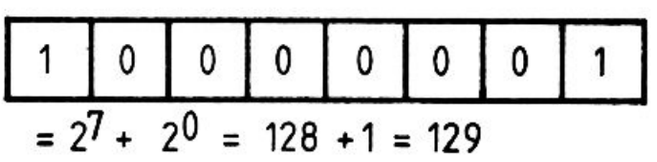


Fig. 2

Nel bit di sinistra si trova un 1 e perciò il suo valore è di 2^7 ovvero 128, nel bit di destra si trova un

1 ed il suo valore è 2^0 ovvero 1. I valori degli altri bit contenuti nel registro sono tutti uguali a 0 e perciò si ha un valore totale di 128+1=129. Torniamo ora all'esempio considerato. Noi vogliamo introdurre un 1 nei primi 4 bit ed uno 0 negli altri 4. In questo modo il registro dati ha selezionato i primi quattro bit come uscite ed i quattro successivi come ingressi.

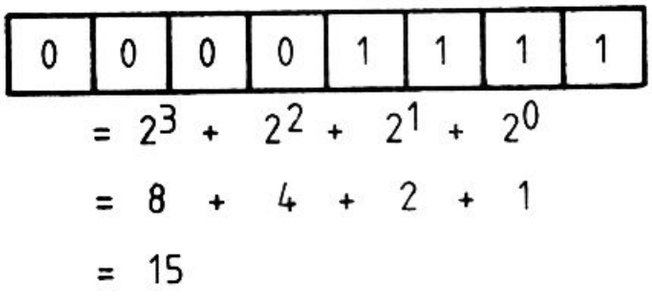


Fig. 3

Introduciamo il valore 15 nel registro PO+3 ed otteniamo così la configurazione richiesta. Come esercizio sono state preparate delle altre configurazioni di bit come numeri decimali e come immagini del registro. Eseguiteli con il programma già noto e confrontate se gli esempi corrispondenti sono in accordo con i contenuti del registro mostrati sullo schermo.
Fate voi stessi delle prove con diverse combinazioni di numeri ed osservate il risultato sullo schermo. In questo modo potrete controllare da soli il vostro lavoro.
Siete ora in grado di modificre a vostro piacimento il registro direzionale. Il prossimo problema consiste nel comunicare una data configurazione di bit alla User-port. Ciò è indispensabile quando voi volete fare in modo che un determinato stato sulla User-port produca un determinato effetto. Se ad esempio una certa azione deve essere effettuata quando i primi due ingressi della User-port sono ad 1 e gli altri sono a 0, potete ottenere ciò con la seguente istruzione:

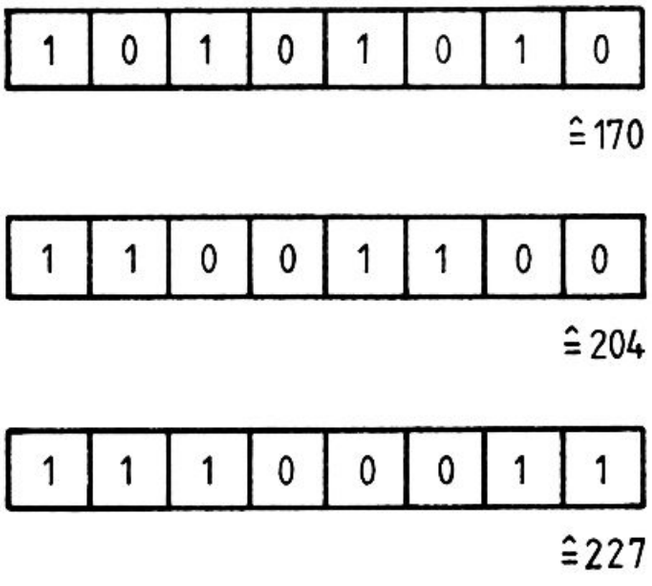


Fig. 4

```
IF PEEK(PO+1)=3 THEN PRINT "AZIONE"
```

Se il registro PO+3 della User-port ha valore 3 allora entrambi i primi due bit sono a 1. Quindi 3 è dato da 2^0 più 2^1. A ciò segue quindi l'azione PRINT.
Per chiedere una determinata configurazione sulla User-port, è quindi sufficiente chiamare il registro corrispondente ed osservare il suo contenuto. Se il contenuto corrisponde al numero che noi desideriamo, allora possiamo provocare un effetto. È invece molto difficile leggere un certo segnale bit dalla User-port. Nella maniera più semplice questo può essere realizzato introducendo semplicemente il numero corrispondente nel rispettivo registro. Se ad esempio il registro di direzione dei dati deve prendere lo stato 11110000 introducete allora semplicemente nel registro 1 il valore 15. Come appare ora la direzione dei dati quando noi introduciamo un 1 nel registro PO+1, dato che la direzione del registro è in realtà in ingresso?

Esempio pratico

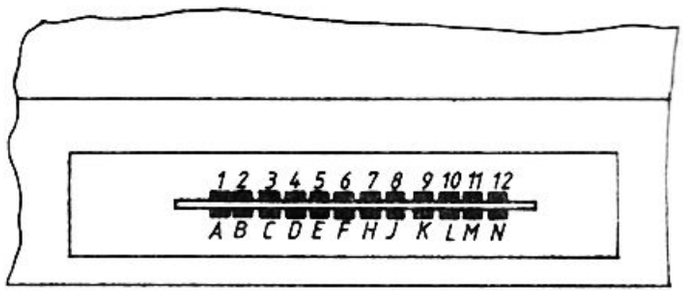
Cancellate ora il vostro programma dalla linea 10 alla linea 80 ed inserite l'esempio seguente. Avviate poi il programma con il comando RUN e vedrete il corrispondente modello dei bit sullo schermo.

```
10 PO=56576
20 POKE PO+3,240
30 GOSUB 1000
40 POKE PO+1,240
50 GOSUB 1000
60 GOTO 50
```

Con il numero 240 abbiamo fissato nella linea 20 la direzione dei dati come uscite per gli ultimi 4 bit e come ingressi per i primi 4. E questo va ancora bene. Nella linea 40 anche la User-port viene caricata con il numero 240. Ora dovrebbe comparire normalmente sullo schermo l'immagine corrispondente. Modificate ora la linea 40 in POKE PO+1,0 ed avviate il programma con RUN. La User-port adesso fornisce per tutti i bit uno 0. Possiamo vedere da questo esempio che la determinazione del contenuto del registro per la User-port (PO+1) rispecchia temporaneamente il programma. Se scriviamo quindi uno 0 in un bit ed il bit

corrispondente è collegato ma come ingresso, questo ingresso ha anch'esso uno 0 in quanto nessun 1 è presente dalla parte esterna dell'ingresso. Un ingresso deve sempre essere comandato dall'esterno e non può essere definito via software. Questo può essere fatto soltanto per il controllo delle uscite. Nel controllo delle uscite, non dovete quindi preoccuparvi dei bit, come invece occorre fare nel controllo degli ingressi. Perciò è possibile un determinato comando come uscita mediante un semplice POKE.

L'ultimo problema consiste ora nel fatto che, con la combinazione di ingressi ed uscite, una determinata configurazione di bit deve essere letta o scritta. Cominciamo con la scrittura della combinazione di bit.
Qui vi sono soltanto due possibilità. La prima é che nel bit si trovi un 1 e si debba mettervi uno 0. La seconda possibilità e che nel bit si trovi uno 0 e si debba mettervi un 1. Cominciamo con l'esempio più semplice: supponiamo che nel bit 7, ovvero a sinistra, si trovi uno 0 ma ci dovrebbe essere un 1 senza che tutti gli altri bit siano cambiati. Perciò dobbiamo considerare il valore del bit che é 128. Con una cosiddetta istruzione OR possiamo ora mettere a 1 questo bit. Prendiamo per così dire il contenuto del registro e lo colleghiamo con un'istruzione OR con il valore 128. Poiché un secondo registro con il contenuto 128 ha un 1 nel bit più a sinistra e 0 in tutti gli altri, dall'OR con il registro della User-port risulta un cambiamento nel bit 7. Tutti gli altri bit non vengono alterati dato che l'OR tra 1 e 0 dà 1 e l'OR tra 0 e 0 dà 0. Con un'istruzione OR possiamo quindi mettere a 1 determinati bit senza che gli altri vengano modificati. Inserite perciò la modifica nella linea 40 ed osservate il risultato sullo schermo. A sinistra il bit é stato messo a 1. Il motivo va ripetuto ancora una volta: si fa un OR tra il registro della User-port ed un registro il cui contenuto é 128. Ciò significa che il bit il cui valore é 128 vale in ogni caso 1 poiché l'OR tra 1 e 0 é 1. Se nella User-port si trova già un 1, allora l'istruzione OR é inutile, comunque risulta ancora un 1. Viene fatto un OR tra tutti gli altri bit e 0 cosicché non si ha nessuna variazione. Modificate nella linea 20 il comando POKE PO+3,240 in POKE PO+3,255:POKE PO+1,0. In tal modo avete comandato tutti i bit come uscite e li avete azzerati. Potete ora provare una determinata combinazione di bit con un'istruzione OR. Osservate il vostro esempio anche sullo schermo.
Il prossimo incarico consiste ora nello scrivere uno 0 in un bit. Se in un bit vi é già un 1 e viene fatto un OR con uno 0, il risultato é ancora 1. Non é quindi possibile risolvere questo problema con un'istruzione OR. Useremo perciò un'istruzione AND. Calcolando l'AND di 1 con 1 il risultato é 1. Si ottiene quindi 1 soltanto quando entrambi i bit sono



1	GND	A	GND
2	+5V	B	FLAG 2
3	RESET	C	PB 0
4	CNT1	D	PB 1
5	SP1	E	PB 2
6	CNT 2	F	PB 3
7	SP 2	H	PB 4
8	PC 2	J	PB 5
9	SER.	K	PB 6
10	9V AC	L	PB 7
11	9V AC	M	PA 2
12	GND	N	GND

Disposizione dei terminali della User-Port

Fig. 5

a 1. 1 AND 0 dà 0 e 0 AND 0 dà ancora 0. Per risolvere il nostro problema dobbiamo perciò unicamente collegare con un AND il bit e uno 0. Il risultato sarà quindi sempre 0. Se la nostra istruzione per cancellare il settimo bit fosse realizzata con AND 128, allora secondo questa istruzione, questo bit sarebbe a zero soltanto quando era già a zero. Noi dobbiamo perciò ribaltare ancora l'istruzione AND. Ciò si ottiene con l'istruzione AND NOT. Modificate quindi la linea 40 in POKE PO+1,PEEK(PO+1) AND NOT 128 ed osservate sullo schermo il contenuto del registro.
Come ulteriore esercizio, cambiate la linea 20 in:

20 POKE PO+3,255:POKE PO+1,255

In questo modo tutte le uscite vengono dapprima messe a 1 e possono ora essere portate a zero con qualsiasi combinazione AND NOT.
Riassumendo si può dire: con un'istruzione OR si può mettere a 1 un bit in un registro. Con una istruzione AND NOT si può mettere a 0 un bit. Viene considerato sempre il bit che, attraverso l'introduzione di un numero, corrisponde al relativo modello di bit nel registro.

```
5 REM INTERFACCIA PER COMPUTER ROBOTICA
1000 REM SIMULATORE USER-PORT
1010 IFPEEK(1185)=85THEN1230
1015 PRINT"■■■
```

CHIP SPECIAL

COMMODORE 64 PROGRAMMI ①
programmi listati, L. 5.000
ISBN 88 7081 153 0
Sono stati inclusi nel volume 27 diversi programmi per questo conosciutissimo home computer: giochi, grafica, musica, organizzazione, scienza e hobby sono i temi trattati. Una breve descrizione iniziale illustra scopo e funzionamento di ciascun programma.
A richiesta ogni programma può essere fornito su cassetta o disco.

COMMODORE 64 PROGRAMMI ②
programmi listati, L. 7.000
ISBN 88 7081 199 9
Giochi, musica, grafica, apprendimento, organizzazione e curiosità varie, sono i temi trattati in questo secondo special per Commodore 64. In totale 26 programmi listati e commentati da una descrizione iniziale che consente all'utilizzatore del computer l'approfondimento delle conoscenze di programmazione. I programmi sono disponibili registrati su cassetta o su dischetto.

COMMODORE 64 PROGRAMMI ③
programmi listati, L. 7.000
ISBN 88 7081 203 0
Questa pubblicazione sul C 64 presenta una serie di nuovi programmi per questo computer tanto popolare. Contiene 17 programmi di giochi, grafica, musica, organizzazione ed hobby. Potete realizzarli per giocare o per risolvere i problemi.

COMMODORE 64 PROGRAMMI ④
programmi listati, L. 7.000
ISBN 88 7081 217 0
Questa edizione speciale risponde al desiderio di buoni programmi applicativi, che servono a facilitare la scrittura nel linguaggio più veloce.
In questo numero infatti troverete un tool-set indispensabile per l'attività di programmazione: un bel programma Monitor e un Assembler a macro particolarmente valido!

CHIP libro club

Mittente

Nome e Cognome

Professione

Via e numero

Città

CAP

Partita IVA

NON AFFRANCARE

Affrancatura a carico del destinatario da addebitarsi sul Conto di Credito n. 3076 presso l'Ufficio Postale di MILANO A. D. (autoriz. Dir. Prov. PP. TT. di Milano n. B/96942 del 27-11-65)

CHIP
Tecniche Nuove
Via Mosca va 46/9
20121 Milano

CHIP libro club

Mittente

Nome e Cognome

Professione

Via e numero

Città

CAP

Partita IVA

NON AFFRANCARE

Affrancatura a carico del destinatario da addebitarsi sul Conto di Credito n. 3076 presso l'Ufficio Postale di MILANO A. D. (autoriz. Dir. Prov. PP. TT. di Milano n. B/96942 del 27-11-65)

CHIP
Tecniche Nuove
Via Mosca va 46/9
20121 Milano

PEEK-POKE
Commodore 64 - Manuale
H. Rügheimer, C. Spanik
L. 8.000
ISBN 88 7081 228 6
Con questo manuale potrete esplorare ed esaminare il campo sconosciuto della memoria del Commodore. Questo libro deve essere una guida, deve spiegare quali siano le locazioni di utilizzo e come impiegarle pienamente. È presente anche una tabella delle parole-chiave e alcune delucidazioni sui termini tecnici.

CHIP libro club

ISBN	TITOLO	PREZZO
2510	Commodore 128. Personal Computer	Lit. 32 000
2138	Commodore Plus/4. Guida all'uso	Lit. 27 500
1581	Commodore VIC 20 Giocavinci - *con cassetta*	Lit. 14 000
1670	Commodore VIC 20 Guida *(libro + cassetta)*	Lit. 19 000
1743	Commodore VIC 20. Il libro dei giochi	Lit. 14 000
8039	Commodore VIC 20. Soft 1 *(cassetta)*	Lit. 8 000
1794	Commodore VIC 20 1° *(cassette)*	Lit. 44 000
1476	Commodore VIC 20 1° listati *(chip special)*	Lit. 6 000
8012	Commodore VIC 20 2° *(cassette)*	Lit. 25 000
2049	Commodore VIC 20 2° listati *(chip special)*	Lit. 7 000
2057	Commodore 16. Manuale d'uso	Lit. 18 000
808X	Commodore 64. Avventuriero *(dischetto)*	Lit. 22 000
1786	Commodore 64. Codice macchina	Lit. 19 000
159X	Commodore 64. Come gestire meglio il C.64	Lit. 22 000
1662	Commodore 64. Guida all'uso	Lit. 10 000
8063	Commodore 64. Hobby elettronica *(dischetto)*	Lit. 22 000
2308	Commodore 64. Musica col C.64	Lit. 16 500
8098	Commodore 64. Party-Nonsense *(dischetto)*	Lit. 22 000
1603	Commodore 64. Programmazione semplice del C.64	Lit. 12 000
8071	Commodore 64. Rompicapo *(dischetto)*	Lit. 22 000
8020	Commodore 64. Soft 8 *(cassetta)*	Lit. 8 000
8055	Commodore 64. Studio e scienza *(dischetto)*	Lit. 22 000
1719	Commodore 64. 1° Programmi registrati su cassetta	Lit. 25 000
8004	Commodore 64. 1° Programmi registrati su dischetto	Lit. 25 000
8160	Commodore 64. 2° Programmi registrati su cassetta	Lit. 25 000
8152	Commodore 64. 2° Programmi registrati su dischetto	Lit. 25 000
8179	Commodore 64. 3° Programmi registrati su cassetta	Lit. 25 000
8187	Commodore 64. 3° Programmi registrati su dischetto	Lit. 25 000
1972	Commodore 64. 40 giochi educativi *(listati + cassetta)*	Lit. 21 000
8047	Commodore 64/Spectrum Soft 2 *(cassetta)*	Lit. 8 000
8101	Commodore 64/Spectrum Soft 3 *(cassetta)*	Lit. 8 000
811X	Commodore 64/Spectrum Soft 4 *(cassetta)*	Lit. 8 000
8136	Commodore 64/Spectrum Soft 5 *(cassetta)*	Lit. 8 000
8195	Commodore 64/Spectrum Soft 6 *(cassetta)*	Lit. 8 000
8209	Commodore 64/Spectrum Soft 7 *(cassetta)*	Lit. 8 000

CHIP libro club

Vogliate inviarmi i seguenti volumi:

Numero	Autore, titolo	Prezzo

PAGAMENTO
☐ assegno bancario/circolare allegato (franco di porto)
☐ contrassegno (+ spese postali)

Data Firma

CHIP libro club

Vogliate inviarmi i seguenti volumi:

Numero	Autore, titolo	Prezzo

PAGAMENTO
☐ assegno bancario/circolare allegato (franco di porto)
☐ contrassegno (+ spese postali)

Data Firma

In questo numero

Simulatore di volo	Il C64 impara a volare
Discoteca	Il C64 come animatore
Party	Il C64 come gioco di luce
Interruttori sonori	Il C64 come maggiordomo
Apriporta	Il C64 come portinaio
Pilotare	Il C64 come pilota
Sorveglianza delle stanze	Il C64 come angelo custode
Fiori	Il C64 come giardiniere
Amplificatore di segnali	Il C64 come controllore ferroviario
Per risparmiare corrente	Il C64 come sorvegliante
Collegamenti	Le prese programmabili
Laboratorio hobbistico	Il C64 come frequenzimetro
Attenuatori	Vi sia luce
Generatori di segnali	Vita da cani
Selezionare, ordinare	Il C64 come robot da tavolo
Alimentazione	Come aumentare la tensione del C64
Dilettanti	Come modificare il mio C64
Interfaccia	Introduzione alla computer-robotica

0704